Coling 2010

23rd International Conference on Computational Linguistics

Proceedings of the Conference
Volume 1

Chu-Ren Huang and Dan Jurafsky

23 – 27 August 2010
Beijing International Convention Center
Beijing, China

Published by
Tsinghua University Press

Block A, Xue Yan Building, Tsinghua University
Beijing, 100084
China

To order hard copies of this proceedings, contact:

Chinese Information Processing Society of China
No.4, Southern Fourth Street
Haidian District, Beijing, 100190
China
Tel: +86-010-62562916
Fax: +86-010-62562916
`cips@iscas.ac.cn`

Sponsorship

The COLING 2010 very gratefully acknowledges the following commitments in sponsorship:

Platinum Sponsors

-National Natural Science Foundation of China

-Department of Language Information Administration, Ministry of Education, PRC

Gold Sponsor

BaiDu

Silver Sponsors

Google

Fujitsu R&D Center CO., LTD.

Microsoft Research

Beijing TRS Information Technology Co., Ltd

Shenyang Globla Envoy software Co.,Ltd.

Supporters

Asian Federation of Natural Language Processing

Institute of Automation
Chinese Academy of Sciences

Institute of Computing Technology
Chinese Academy of Sciences

Institute of Software
Chinese Academy of Sciences

Harbin Institute of Technology

Peking University

Tsinghua University

Preface

You will find in this volume papers from the 23rd International Conference on Computational Linguistics (COLING 2010) held in Beijing, China on August 23-27, 2010 under the auspices of the International Committee on Computational Linguistics (ICCL), and organized by the Chinese Information Processing Society (CIPS) of China. For this prestigious natural language processing conference to be held in China is a significant event for computational linguistics and for colleagues in China, demonstrating both the maturity of our field and the development of academic areas in China.

COLING started as a friendly gathering in New York in 1965, and has grown steadily since. Yet COLINGs aspiration to be a different conference remains the same. COLING strives to maintain its key qualities of embracing different theories and encouraging young scholars in spite of its growing size. A new component introduced at COLING 2010 underlines this quality. A RefreshINGenious (RING) session, organized by Aravind Joshi, our General Chair, allows new and un-orthodox ideas to be presented before they are fully developed in order to generate more discussion and stimulate other new ideas. We hope that this can become an important feature of COLING in the future.

The 155 oral papers included in the hardcopy proceedings published by Tsinghua University Press, as well as the 334 papers included in the electronic proceedings (the same 155 oral papers plus 179 poster papers) are selected from among 815 effective submissions among the more than 840 submissions received. The very selective acceptance rate of 19.02% for oral presentations (155/815 submissions) indicates the extremely high quality of the papers. An additional 21.96% (179/815) are selected for poster presentations to bring the overall acceptance rate to 40.98% (334/815).

We would like to thank the program committee area chairs for their dedicated and efficient review work, and our 738 reviewers for giving us very high quality reviews with a very short turnaround time, allowing us to maintain both the review quality and schedule even given the extraordinary number of submissions. Of course we thank the authors of the 840 papers for submitting their labor of love to COLING. Although we were only able to accept a minority of the submitted papers, we do hope that all authors and reviewers benefit from this process of indirect dialogue. We are especially grateful to the incredibly hard-working team of Stanford volunteers Jenny Finkel, Adam Vogel, and Mengqiu Wang, and HIT volunteers Sam Liang and Lemon Liu, who provided timely and efficient support for the two program chairs at every step of the review and publication processes.

Last but not least, we would like to thank the people who made COLING 2010 and this volume possible. We thank local arrangement committee co-chairs Professor Chengqing Zong and Professor Le Sun for their tireless work which will make COLING-2010 a sure success. Our special appreciation goes to the Chinese Information Processing Society (CIPS) and Professor Youqi Cao for their generous support as the COLING 2010 organizer. Lastly, Professor Qin Lu and Professor Tiejun Zhao should be recognized for their meticulous preparation for editing and publication, which brought this volume to reality.

Chu-Ren Huang and Dan Jurafsky,
COLING 2010 Program Committee Co-chairs

July 8, 2010

COLING 2010 is organized by the Chinese Information Processing Society of China (**CIPS**) and under the auspices of the International Committee on Computational Linguistics (**ICCL**).

General Chair:

Aravind K .Joshi (University of Pennsylvania)

Program Chairs:

Chu-Ren Huang (The Hong Kong Polytechnic University)
Dan Jurafsky (Stanford University)

Advisors to Organizing Committee:

Youqi Cao (The Chinese Information Processing Society of China)
Zhendong Dong (The Chinese Information Processing Society of China)
Changning Huang (Microsoft Research Asia)
Sheng Li (Harbin Institute of Technology)
TianshunYao (Northeastern University)
Shiwen Yu (Peking University)
Zhiwei Feng (Institute of Applied Linguistics, Ministry of Education)
Kaiying Liu (Shanxi University)

OrganizationChairs:

Chengqing Zong (Institute of Automation, Chinese Academy of Sciences)
Le Sun (Institute of Software, Chinese Academy of Sciences)

Publication Chairs:

Qin Lu (The Hong Kong Polytechnic University)
Tiejun Zhao (Harbin Institute of Technology)

Tutorial Chairs:

Dan Gildea (University of Rochester)
Xuanjing Huang (Fudan University)

Workshop Chairs:

Noah Smith (Carnegie Mellon University)
Takenobu Tokunaga (Tokyo Institute of Technology)
Haifeng Wang (BaiDu)

Publicity Chairs:

Hal Daumé III (University of Utah)
Bin Wang (Institute of Computing Technology, Chinese Academy of Sciences)
Minghui Dong (Institute for Infocomm Research)
Monica Monachini (Institute for Computational Linguistics)

Aline Villavicencio (Federal University of Rio Grande do Sul)

Sponsorship Chairs:

Tingting He (Huazhong Normal University)
Hinrich Schütze (University of Stuttgart)
Key-Sun Choi (Korea Advanced Institute of Science and Technology)
Rion Snow (Stanford University)
Takehito Utsuro (University of Tsukuba)
Renata Vieira (Pontifical Catholic University of Rio Grande do Sul)
Hao Yu (Fujitsu(China) R&D Center)

Demo Chairs:

Ting Liu (Harbin Institute of Technology)
Yang Liu (The University of Texas at Dallas)

Program Committee Members/AreaChairs:

Nianwen Xue (Brandeis University)
Rajeev Sangal (India Institute of Information Technology)
Roger Levy (University of California, San Diego)
Justine Cassell (Northwestern University)
Caroline Sporleder (Saarland University)
Gary Geunbae Lee (Pohang University of Science and Technology)
Rosie Jones (Yahoo! Research)
Nicoletta Calzolari (Istituto di Linguistica Computazionale)
Chris Callison-Burch (Johns Hopkins University)
Qun Liu (Institute of Computing Technology)
Pierre Isabelle (National Research Council of Canada)
Sadao Kurohashi (Kyoto University)
Sebastian Padó (Universität Stuttgart)
Takenobu Tokunaga (Tokyo Institute of Technology)
Bo Pang (Yahoo! Research)
Haizhou Li (Institute for Infocomm Research)
David A Smith (University of Massachusetts, Amherst)
Donia Scott (University of Sussex)
Emily Bender (University of Washington)
Ming Zhou (Microsoft Research Asia)

Organization Committee Members:

Juanzi Li (Tsinghua University)
KWONG Olivia (Hong Kong City University)
Bin Sun (Peking University)
Houfeng Wang (Peking University)
Xiaojie Wang (Beijing University of Posts and Telecommunications)
Endong Xun (Beijing Language and Culture University)
Erhong Yang (Beijing Language and Culture University)
Jun Zhao (Institute of Automation, Chinese Academy of Sciences)
Jingbo Zhu (Northeastern University)

Program Committee Members/Reviewers:

Rob Abbott
Takeshi Abekawa
Omri Abend
Steven Abney
Eytan Adar
Guadalupe Aguado
Khalid Al-Kofahi
Cecilia Ovesdotter Alm
Omar Alonso
Bharat Ram Ambati
Massih-Reza Amini
Xiangdong An
Sophia Ananiadou
Jaime Arguello
Masayuki Asahara
Nicholas Asher
Tania Avgustinova
Necip Fazil Ayan
Lakshmi Bai
Jason Baldridge
Timothy Baldwin
Carmen Banea
Srinivas Bangalore
Colin Bannard
Ken Barker
Marco Baroni
Regina Barzilay
John Bateman
Beata Beigman Klebanov
Daisuke Bekki
Nria Bel
Yinon Bentor
Sabine Bergler
Aditya Bhargava
Pushpak Bhattacharyya
Timothy Bickmore
Klinton Bicknell
Alexandra Birch
Philippe Blache
Alan Black
John Blitzer
Michael Bloodgood
Phil Blunsom
Ondrej Bojar
Gemma Boleda
Francis Bond
Johan Bos
Marisa Boston
Pierrette Bouillon
Julien Bourdaillet
Jordan Boyd-Graber
S. R. K. Branavan
Antonio Branco
Eric Breck
Chris Brew
Sabine Buchholz
Stefan Buettcher
Hung Bui
Razvan Bunescu
Aljoscha Burchardt
Stephan Busemann
Miriam Butt
William Byrne
Lynne Cahill
Jamie Callan
Bin Cao
Yunbo Cao
Guiseppe Carenini
Jean Carletta
Marine Carpuat
Xavier Carreras
John Carroll
Ben Carterette
Francisco Casacuberta
Steve Cassidy
Eric Castelli
Alexandru Ceausu
Nick Cercone
Jeong-Won Cha
Vineet Chaitanya
Yllias Chali
Nate Chambers
Baobao Chang
Pi-Chuan Chang
Wanxiang Che
Ciprian Chelba
Aitao Chen
Bin Chen
Boxing Chen
Chien Chin Chen
Harr Chen
Hsin-Hsi Chen
Jinying Chen
Keh-Jiann Chen
Wenliang Chen
Xiao Chen
Charibeth K. Cheng
Colin Cherry
Tee Kiah Chia
David Chiang
Yejin Choi
Kenneth Church
Massimiliano Ciaramita
Philipp Cimiano
Shay Cohen
Kevyn Collins-Thompson
John Conroy
Bonaventura Coppola
Ed Cormany
Dan Cristea
Bruce Croft
Dick Crouch
Montse Cuadros
Silviu-Petru Cucerzan
Oliver Čulo
Aron Culotta
Beatrice Daille
Hercules Dalianis
Sandipan Dandapat
Hoa Dang
Dana Dannells
Dipanjan Das
Sajib Dasgupta
Hal Daume
Dmitry Davidov
Adria de Gispert
Eric De La Clergerie
Valeria de Paiva
Maarten de Rijke
Thierry Declerck
Vera Demberg
Steve DeNeefe
John DeNero
Pascal Denis
Mona Diab
Georgiana Dinu
Pinar Donmez
Iustin Dornescu
Ascander Dost
Antoine Doucet
Doug Downey
Markus Dreyer
Gregory Druck
Xiangyu Duan
Amit Dubey
Kevin Duh
Michael Dukes
Chris Dyer
Marc Dymetman
Markus Egg

Koji Eguchi
David Eichmann
Andreas Eisele
Jacob Eisenstein
Jason Eisner
Noemie Elhadad
Charles Elkan
Micha Elsner
Alexandre entiev
Katrin Erk
Andrea Esuli
Richard Evans
Roger Evans
Patrick Fan
Alex Fang
Benoit Favre
Marcello Federico
Christiane Fellbaum
Katja Filippova
Radu Florian
Sandiway Fong
George Foster
Gil Francopoulo
Stefan Frank
Alex Fraser
Marjorie Freedman
Maria Fuentes
Hagen Fuerstenau
Atsushi Fujii
Sumio Fujita
Pascale Fung
Ryan Gabbard
Evgeniy Gabrilovich
Karthik Gali
Michel Galley
Michael Gamon
Kuzman Ganchev
Jianfeng Gao
Albert Gatt
Eric Gaussier
Ruifang Ge
Byron Georgantopoulos
Pablo Gervas
Sanjukta Ghosh
Daniel Gildea
Dan Gillick
Jesus Gimenez
Kevin Gimpel
Roxana Girju
Sharon Goldwater
Tianxia Gong
Julio Gonzalo
Cyril Goutte
Brigitte Grau
Stephan Greene
Mark Greenwood
Gregory Grefenstette
Ralph Grishman
Cecile Grivaz
Jiafeng Guo
Ben Hachey
Aria Haghighi
Udo Hahn
Jan Hajic
Eva Hajicova
Dilek Hakkani-tur
John Hale
David Hall
Greg Hanneman
Max Harper
Chikara Hashimoto
Ahmed Hassan
Claudia Hauff
Yoshihiko Hayashi
Ben He
daan he
Daqing He
Xiaodong He
Yulan He
John Henderson
Iris Hendrickx
Amac Herdagdelen
Ulf Hermjakob
Raquel Hervas
Dirk Heylen
Andrew Hickl
Graeme Hirst
Jerry Hobbs
Katja Hofmann
Steven Chu Hong Hoi
Kristy Hollingshead
Mark Hopkins
Eric Horvitz
Veronique Hoste
Yunhua Hu
Jimmy Huang
XuanJing Huang
Yifen Huang
Yun Huang
Matt Huenerfauth
Juan Huete
Sarmad Hussain
Rebecca Hwa
Gonzalo Iglesias
Steven Ikier
Diana Inkpen
Kentaro Inui
Elena Irimia
Amy Isard
Abe Ittycheriah
Tatsuya Izuha
Adam Jatowt
Jiwoon Jeon
Heng Ji
Sittichai Jiampojamarn
Daxin Jiang
Hongfei Jiang
Jiepu Jiang
Jing Jiang
Long Jiang
Wenbin Jiang
Valentin Jijkoun
Richard Johansson
Howard Johnson
Rie Johnson
Kristiina Jokinen
Doug Jones
Rosie Jones
Hanmin Jung
Vijay K. Shanker
Min-Yen Kan
Hiroshi Kanayama
Damianos Karakos
Nikiforos Karamanis
Rohit Kate
Tsuneaki Kato
Daisuke Kawahara
Tatsuya Kawahara
Junichi Kazama
Shahram Khadivi
Chloe Kiddon
Jin-Dong Kim
Min Kim
Chunyu Kit
Kevin Knight
Youngjoong Ko
Philipp Koehn
Rob Koeling
Dimitrios Kokkinakis
Greg Kondrak

Moshe Koppel
Valia Kordoni
Lili Kotlerman
Eric Kow
Zornitsa Kozareva
Emiel Kramer
Steven Krauwer
Gerhard Kremer
Canasai Kruengkrai
Lun-Wei Ku
Taku Kudo
Jonas Kuhn
Roland Kuhn
Peter Khnlein
Amba Kulkarni
Ravi Kumar
Shankar Kumar
A Kumaran
Oren Kurland
K. L. Kwok
Olivia Kwong
Wai Lam
Man Lan
Philippe Langlais
Francois Lareau
Martha Larson
Alex Lascarides
Alberto Lavelli
Alon Lavie
Florian Laws
Guy Lebanon
Changki Lee
John Lee
Joo Young Lee
Lillian Lee
Seungwoo Lee
Tan Lee
Jochen Leidner
Yves Lepage
Kevin Lerman
James Lester
Gregor Leusch
Gina-Anne Levow
Hang Li
Jiye Li
Lei Li
Mu Li
Shiqi Li
Shoushan Li
Wei Li
Wen Jie Li
Zhifei Li
Dekang Lin
Lucian Vlad Lita
Diane Litman
Bing Liu
Shui Liu
Ting Liu
Xiaohua Liu
Yan Liu
Yang Liu
Yang Liu
Yupeng Liu
Zhanyi Liu
Zhaopengx Liu
Anna Lobanova
Qiu Long
Adam Lopez
Qin Lu
Yue Lu
Harald Lngen
Xiaoqiang Luo
Tomoya Lwakura
Bin Ma
Qing Ma
Yanjun Ma
Wolfgang Macherey
Nitin Madnani
Hala Maghout
B. Mallikarjun
Gideon Mann
Daniel Marcu
Mitch Marcus
Joseph Mariani
Katja Markert
Erwin Marsi
M. Antonia Marti Antonin
Jean-Claude MARTIN
Andre Martins
Yuval Marton
Yuji Masumoto
Shigeki Matsubara
Tomoko Matsui
Yuji Matsumoto
Takuya Matsuzaki
Evgeny Matusov
Mausam
Arne Mauser
Marshall Mayberry
Diana McCarthy
David McClosky
Ryan McDonald
Tara McIntosh
Kathy McKeown
Susan McRoy
Yashar Mehdad
Qiaozhu Mei
Chris Mellish
Amlia Mendes
Helen Meng
Donald Metzler
Haitao Mi
Jun Miao
Lukas Michelbacher
Jeffrey Micher
Eleni Miltsakaki
David Mimno
Zhaoyan Ming
Shachar Mirkin
Jeff Mitchell
Vibhu Mittal
Yusuke Miyao
Marie-Francine Moens
Dan Moldovan
Diego Moll Aliod
Christof Monz
Raymond J. Mooney
Bob Moore
Roser Morante
Louis-Philippe Morency
Alessandro Moschitti
Isabelle Moulinier
Animesh Mukherjee
Stefan Mller
Art Munson
Dragos Munteanu
Masaki Murata
Vanessa Murdock
Smaranda Muresan
Gabriel Murray
Pradeep Muthukrishnan
Sobha Nair
Testuji Nakagawa
Vivi Nastase
Martina Naughton
Roberto Navigli
Mark-Jan Nederhof
Vasek Nemcik
Ani Nenkova
Hwee Tou Ng

Vincent Ng
Huyen Nguyen Thi Minh
Patrick Nguyen
Alex Niculescu-Mizil
Jian-Yun Nie
Rodney Nielsen
Takashi Ninomiya
Toyoaki Nishida
Cheng Niu
Zheng-Yu Niu
Diarmuid Saghdha
Franz Och
Michael ODonnell
Stephan Oepen
Naoki Okazaki
Lilja Ovrelid
Derya Ozkan
Ulrike Pado
Alexis Palmer
Sinno Jialin Pan
Patrick Pantel
Simone Paolo Ponzetto
Jong Park
Kristen Parton
Marius Pasca
Siddharth Patwardhan
Michael Paul
Soma Paul
Matthias Paulik
Yves Peirsman
Gerald Penn
Marco Pennacchiotti
Bhaskararao Peri
Aina Peris
Wim Peters
Kay Peterson
Slav Petrov
Daniele Pighin
Prasad Pingali
Stelios Piperidis
Guillaume Pitel
Paul Piwek
Luiz Pizzato
Massimo Poesio
Hoifung Poon
Ana-Maria Popescu
Maja Popovic
Christopher Potts
Richard Power
Rashmi Prasad
Laurent Prevot
Matthew Purver
Haoliang Qi
Toh Zhi Qiang
Tao Qin
Yang Qu
Chris Quirk
Bhiksha Raj
S. Rajendran
Dan Ramage
S.V. Ramanan
P. V. S. Rambabu
Owen Rambow
Delip Rao
Ari Rappoport
Paul Rayson
Michaela Regneri
Nils Reiter
Norbert Reithinger
Giuseppe Riccardi
Sebastian Riedel
German Rigau
Hae-Chang Rim
Lucia Rino
Hammam Riza
Horacio Rodrłguez
Gerhard Rolletschek
Lorenza Romano
Mathias Rossignol
Antti-Veikko Rosti
Michael Roth
Alex Rudnick
Marta Ruiz
Kenji Sagae
Horacio Saggion
Harabagiu Sanda
Baskaran Sankaran
Ratna Sanyal
Murat Saraclar
Sudeshna Sarkar
Manabu Sassano
Sekine Satoshi
Roser Sauri
Helmut Schmid
William Schuler
Sabine Schulte im Walde
Ineke Schuurman
Lane Schwartz
Wolfgang Seeker
sekine Sekine
Stephanie Seneff
Violeta Seretan
Hendra Setiawan
Chirag Shah
Dipti Sharma
Dou Shen
Libin Shen
Wade Shen
Kiyoaki Shirai
Eyal Shnarch
Advaith Siddharthan
Candy Sidner
Khalil Simaan
Michel Simard
Kiril Simov
Anil Kumar Singh
Smriti Singh
Samar Sinha
Sharon Small
Jason Smith
Nathaniel Smith
Matthew Snover
Benjamin Snyder
Stephen Soderland
Swapna Somasundaran
Yan Song
Young-In Song
Virach Sornlertlamvanich
Lucia Specia
Jennifer Spenader
Valentin Spitkovsky
Richard Sproat
Ed Stabler
Manfred Stede
Mark Steedman
Amanda Stent
Mark Stevenson
Matthew Stone
Svetlana Stoyanchev
Veselin Stoyanov
Kristina Striegnitz
Michael Strube
Tomek Strzalkowski
Jian Su
Jinsong Su
Keh-Yih Su
Nam Kim Su
Rajen Subba
K. V. Subbarao
L. V. Subramaniam

Zhifang Sui
Eiichiro Sumita
Congkai Sun
Jun Sun
Maosong Sun
Hisami Suzuki
Gyorgy Szarvas
Hiroya Takamura
Koichi Takeuchi
David Talbot
Chew Lim Tan
Jie Tang
Chua Tat-Seng
Paul Tepper
Simone Teufel
Stefan Thater
Mariet Theune
Paul Thomas
Vu Thuy
Joerg Tiedemann
Christoph Tillmann
Ivan Titov
Katrin Tomanek
Yoichi Tomiura
Sara Tonelli
Kentaro Torisawa
Kristina Toutanova
Roy Tromble
Huihsin Tseng
Shu-Chuan Tseng
Benjamin Tsou
Yuta Tsuboi
Jun-ichi Tsujii
Koji Tsukamoto
Yoshimasa Tsuruoka
Dan Tufis
Gokhan Tur
Kiyotaka Uchimoto
Yuya Unno
Lonneke Van der Plas
ielka van der sluis
Josef van Genabith
Gertjan van Noord
Lucy Vanderwende
Sebastian Varges
Sriram Venkatapathy
Marc Verhagen
Yannick Versley
Cristina Vertan
Marta Vila
David Vilar
Aline Villavicencio
Stephan Vogel
Clare Voss
Piek Vossen
Marilyn Walker
Hanna Wallach
Xiaojun Wan
Wenting Wang
Haifeng Wang
Hsin-Min Wang
Kai Wang
Rui Wang
Wei Wang
Xuanhui Wang
Leo Wanner
Nigel Ward
Taro Watanabe
Yotaro Watanabe
Nick Webb
Bonnie Webber
Eric Wehrli
Kilian Weinberger
David Weir
Michael White
Michael Wick
Jan Wiebe
Yorick Wilks
Theresa Wilson
Shuly Wintner
Andreas Witt
Peter Wittek
Magda Wolska
Chung-Hsien Wu
Dan Wu
Dekai Wu
Hua Wu
Xiaoyun Wu
Yunfang Wu
Fei Xia
Rui Xia
Xinyan Xiao
Deyi Xiong
Peng Xu
Ruifeng Xu
Bert Xue
Yongxin Yan
Muyun Yang
Qiang Yang
Ainur Yessenalina
Bei Yu
Jianxing Yu
Kun Yu
Mo Yu
Yisong Yue
Zdenek Zabokrtsky
Taras Zagibalov
Omar Zaidan
Roberto Zanoli
Fabio Zanzotto
Alessandra Zarcone
Richard Zens
Torsten Zesch
Luke Zettlemoyer
Hao Zhang
Hui Zhang
Jiajun Zhang
Min Zhang
Ruiqiang Zhang
Yujie Zhang
Ziqi Zhang
Hai Zhao
Jun Zhao
Bowen Zhou
Guodong Zhou
Yiping Zhou
Conghui Zhu
Jerry Zhu
Tao Zhuang
Heike Zinsmeister
Imed Zitouni
Andreas Zollmann
Chengqing Zong
Ingrid Zukerman

Table of Contents

Author Index

Testing SDRT's Right Frontier

Stergos D. Afantenos and **Nicholas Asher**
Institut de recherche en informatique de Toulouse (IRIT),
CNRS, Université Paul Sabatier
{stergos.afantenos, nicholas.asher}@irit.fr

Abstract

The Right Frontier Constraint (RFC), as a constraint on the attachment of new constituents to an existing discourse structure, has important implications for the interpretation of anaphoric elements in discourse and for Machine Learning (ML) approaches to learning discourse structures. In this paper we provide strong empirical support for SDRT's version of RFC. The analysis of about 100 doubly annotated documents by five different naive annotators shows that SDRT's RFC is respected about 95% of the time. The qualitative analysis of presumed violations that we have performed shows that they are either click-errors or structural misconceptions.

1 Introduction

A cognitively plausible way to view the construction of a discourse structure for a text is an incremental one. Interpreters integrate discourse constituent n into the antecedently constructed discourse structure D for constituents 1 to $n-1$ by linking n to some constituent in D with a discourse relation. SDRT's Right Frontier Constraint (RFC) (Asher, 1993; Asher and Lascarides, 2003) says that a new constituent n cannot attach to an arbitrary node in D. Instead it must attach to either the last node entered into the graph or one of the nodes that dominate this last node. Assuming that the last node is usually found on the right of the structure, this means that the nodes available for attachment occur on the *right frontier* (RF) of the discourse *graph* or SDRS.

Researchers working in different theoretical paradigms have adopted some form of this constraint. Polanyi (1985; 1988) originally proposed the RFC as a constraint on antecedents to anaphoric pronouns. SDRT generalizes this to a condition on all anaphoric elements. As the attachment of new information to a contextually given discourse graph in SDRT involves the resolution of an anaphoric dependency, RFC furnishes a constraint on the attachment problem. (Webber, 1988; Mann and Thompson, 1987; 1988) have also adopted versions of this constraint. But there are important differences. While SDRT and RST both take RFC as a constraint on all discourse attachments (in DLTAG, in contrast, anaphoric discourse particles are not limited to finding an antecedent on the RF), SDRT's notion of RF is substantially different from that of RST's or Polanyi's, because SDRT's notion of a RF depends on a 2-dimensional discourse graph built from *coordinating* and *subordinating* discourse relations. Defining RFC with respect to SDRT's 2-dimensional graphs allows the RF to contain discourse constituents that do not include the last constituent entered into the graph (in contrast to RST). SDRT also allows for multiple attachments of a constituent to the RFC.

SDRT's RFC has important implications for the interpretation of various types of anaphoric elements: tense (Lascarides and Asher, 1993), ellipsis (Hardt et al., 2001; Hardt and Romero, 2004; Asher, 2007), as well as pronouns referring to individuals and abstract entities (Asher, 1993; Asher and Lascarides, 2003). The RFC, we believe, will also benefit ML approaches to learning discourse structures, as a constraint limiting the search space for possible discourse attachments. Despite its importance, SDRT's RFC has never been empirically validated, however. We present evidence in this paper providing strong empirical support for SDRT's version of the constraint. We have chosen to study SDRT's notion of a RF, because of SDRT's greater expressive power over RST (Danlos, 2008), the greater generality of SDRT's defi-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 1–9,
Beijing, August 2010

nition of RFC, and because of SDRT's greater theoretical reliance on the constraint for making semantic predictions. SDRT also makes theoretically clear why the RFC should apply to discourse relation attachment, since it treats discourse structure construction as a dynamic process in which all discourse relations are essentially anaphors. The analysis of about 100 doubly annotated documents by five different naive annotators shows that this constraint, as defined in SDRT, is respected about 95% of the time. The qualitative analysis of the presumed violations that we have performed shows that they are either click-errors or structural misconceptions by the annotators.

Below, we give a formal definition of SDRT's RFC; section 3 explains our annotation procedure. Details of the statistical analysis we have performed are given in section 4, and a qualitative analysis is provided in section 5. Finally, section 6 presents the implications of the empirical study for ML techniques for the extraction of discourse structures while sections 7 and 8 present the related work and conclusions.

2 The Right Frontier Constraint in SDRT

In SDRT, a discourse structure or SDRS (Segmented Discourse Representation Structure) is a tuple $< A, \mathcal{F}, \text{LAST} >$, where A is the set of labels representing the discourse constituents of the structure, LAST $\in A$ the last introduced label and $\mathcal{F}$ a function which assigns each member of A a well-formed formula of the SDRS language (defined (Asher and Lascarides, 2003, p 138)). SDRSs correspond to λ expressions with a continuation style semantics. SDRT distinguishes coordinating and subordinating discourse relations using a variety of linguistic tests (Asher and Vieu, 2005),[1] and isolates structural relations (Parallel and Contrast) based on their semantics.

The RF is the set of available attachment points to which a new utterance can be attached. What this set includes depends on the discourse relation used to make the attachment. Here is the definition from (Asher and Lascarides, 2003, p 148).

> Suppose that a constituent β is to be attached to a constituent in the SDRS with a discourse relation other than Parallel or Contrast. Then the available attachment points for β are:
>
> 1. The label $\alpha = \text{LAST}$;
> 2. Any label γ such that:
>   (a) *i-outscopes*(γ, α) (*i.e.* $R(\delta, \alpha)$ or $R(\alpha, \delta)$ is a conjunct in $\mathcal{F}(\gamma)$ for some R and some δ); or
>   (b) $R(\gamma, \alpha)$ is a conjunct in $\mathcal{F}(\lambda)$ for some label λ, where R is a subordinating discourse relation.
>
>   We gloss this as $\alpha < \gamma$.
> 3. Transitive Closure:
>   Any label γ that dominates α through a sequence of labels $\gamma_1, \gamma_2, \ldots \gamma_n$ such that $\alpha < \gamma_1 < \gamma_2 < \ldots \gamma_n < \gamma$

We can represent an SDRS as a graph $\mathcal{G}$, whose nodes are the labels of the SDRSs constituents and whose typed arcs represent the relations between them. The nodes available for attachment of a new element β in $\mathcal{G}$ are the last introduced node LAST and any other node dominating LAST, where the notion of domination should be understood as the transitive closure over the arrows given by *subordinating* relations or those holding between a complex segment and its parts. Subordinating relations like *Elaboration* extend the vertical dimension of the graph, whereas coordinating relations like *Narration* expand the structure horizontally. The graph of every SDRS has a unique top label for the whole structure or formula; however, there may be multiple $<$ paths defined within a given SDRS, allowing for multiple parents, in the terminology of (Wolf and Gibson, 2006). Furthermore, SDRT allows for multiple arcs between constituents and attachments to multiple constituents on the RFC, making for a very rich structure.

SDRT's RFC is restricted to non-structural relations, because structural relations postulate a partial isomorphism from the discourse structure of the second constituent to the discourse structure of the first, which provides its own attachment possibilities for subconstituents of the two related structures (Asher, 1993). Sometimes such parallelism or contrast, also known as *discourse subordination* (Asher, 1993), can be enforced in a long

[1]The subordinating relations of SDRT are currently: Elaboration (a relation defined in terms of the main eventualities of the related constituents), Entity-Elaboration (E-Elab(a,b) iff b says more about an entity mentioned in a that is not the main eventuality of a) Comment, Flashback (the reverse of Narration), Background, Goal (intentional explanation), Explanation, and Attribution. The coordinating relations are: Narration, Contrast, Result, Parallel, Continuation, Alternation, and Conditional, all defined in Asher and Lascarides (2003).

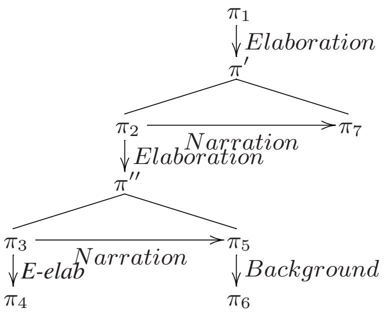
distance way by repeating the same wording in the two constituents.

RFC has the name it does because the segments that belong on this set (the γs in the above definition) are typically nodes on a discourse graph which are geometrically placed at the RF of the graph. Consider the following example embellished from Asher and Lascarides (2003):

(1) (π_1) John had a great evening last night. (π_2) He first had a great meal at Michel Sarran. (π_3) He ate profiterolles de foie gras, (π_4) which is a specialty of the chef. (π_5) He had the lobster, (π_6) which he had been dreaming about for weeks. (π_7) He then went out to a several swank bars.

The graph of the SDRS for 1 looks like this:

(2)

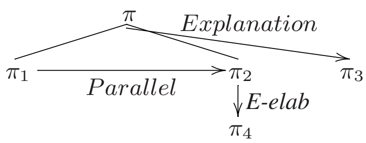


where π' and π'' represent complex segments. Given that the last introduced utterance is represented by the node π_7, the set of nodes that are on the RF are π_7 (LAST), π' (the complex segment that includes π_7) and π_1 (connected via a subordinating relation to π'). All those nodes are geometrically placed at the RF of the graph.

SDRT's notion of a RF is more general than RST's or DLTAG's. First, SDRSs can have complex constituents with multiple elements linked by coordinate relations that serve as arguments to other relations, thus permitting instances of *shared structure* that are difficult to capture in a pure tree notation (Lee et al., 2008). In addition, in RST the RF picks out the *adjacent* constituents, LAST and complex segments including LAST. Contrary to RST, SDRT, as it uses 2-dimensional graphs, predicts that an available attachment point for π_7 is the non local and non adjacent π_2, which is distinct from the complex constituent consisting of π_2 to π_6.[2] This difference is crucial to the interpretation of the Narration: Narration claims a sequence of two events; making the complex constituent (essentially a sub-SDRS) an argument of Narration, as RST does, makes it difficult to recover such an interpretation. Danlos's (2008) interpretation of the Nuclearity Principle provides an interpretation of the Narration([2-4],5) that is equivalent to the SDRS graph above.[3] But even an optional Nuclearlity Principle interpretation won't help with discourse structures like (2) where the backgrounding material in π_4 and the commentary in π_6 do not and cannot figure as part of the Elaboration for semantic reasons. In our corpus described below, over 20% of the attachments were non adjacent; *i.e.* the attachment point for the new material did not include LAST.

A further difference between SDRT and other theories is that, as SDRT's RFC is applied recursively over complex segments within a given SDRS, many more attachment points are available in SDRT. E.g., consider the SDRS for this example, adapted from (Wolf and Gibson, 2006):

(3) (π_1) Mary wanted garlic and thyme. (π_2) She also needed basil. (π_3) The recipe called for them. (π_4) The basil would be hard to come by this time of year.



Because π is the complex segment consisting of π_1 and π_2, attachment to π with a subordinating discourse relation permits attachment π's open constituents as well.[4]

3 Annotated Corpus

Our corpus comes from the discourse structure annotation project ANNODIS[5] which represents an on going effort to build a discourse graph bank for French texts with the two-fold goal of testing various theoretical proposals about discourse

[2]The 2-dimensionality of SDRSs also allows us to represent many examples with Elaboration that involve crossing dependencies in Wolf and Gibson's (2006) representation without violation of the RFC.

[3]Baldridge et al. (2007), however, show that the Nuclearity Principle does not always hold.

[4]This part of the RFC was not used in (Asher and Lascarides, 2003).

[5]http://w3.erss.univ-tlse2.fr/annodis

structure and providing a seed corpus for learning discourse structures using ML techniques. ANNODIS's annotation manual provides detailed instructions about the segmentation of a text into Elementary Discourse Units (EDUs). EDUs correspond often to clauses but are also introduced by frame adverbials,[6] appositive elements, correlative constructions (*[the more you work,] [the more you earn]*), interjections and discourse markers within coordinated VPs *[John denied the charges] [but then later admitted his guilt]*. Appositive elements often introduce *embedded* EDUs; e.g., *[Jim Powers, [President of the University of Texas at Austin], resigned today.]*, which makes our segmentation more fine-grained than Wolf and Gibson's (2006) or annotation schemes for RST or the PDTB.

The manual also details the meaning of discourse relations but says nothing about the structural postulates of SDRT. For example, there is no mention of the RFC in the manual and very little about hierarchical structure. Subjects were told to put whatever discourse relations from our list above between constituents they felt were appropriate. They were also told that they could group constituents together whenever they felt that as a whole they jointly formed the term of a discourse relation. We purposely avoided making the manual too restrictive, because one of our goals was to examine how well SDRT predicts the discourse structure of subjects who have little knowledge of discourse theories.

In total 5 subjects with little to no knowledge of discourse theories that use RFC participated in the annotation campaign. Three were undergraduate linguistics students and two were graduate linguistics students studying different areas. The 3 undergraduates benefitted from a completed and revised annotation manual. The two graduate students did their annotations while the annotation manual was undergoing revisions. All in all, our annotators doubly annotated about 100 French newspaper texts and *Wikipedia* articles. Subjects first segmented each text into EDUs, and then they were paired off and compared their segmentations, resolving conflicts on their own or via a supervisor. The annotation of the discourse relations was performed by each subject working in isolation. ANNODIS provided a new state of the art tool, GLOZZ, for discourse annotation for the three undergraduates. With GLOZZ annotators could isolate sections of text corresponding to several EDUs, and insert relations between selected constituents using the mouse. Though it did portray relations selected as lines between parts of the text, GLOZZ did not provide a discourse graph or SDRS as part of its graphical interface. The representation often yielded a dense number of lines between segments that annotators and evaluators found hard to read. The inadequate interline spacing in GLOZZ also contributed to certain number of click errors that we detail below in the paper. The statistics on the number of documents, EDUs and relations provided by each annotator are in table 1.

annotator	*# Docs*	*# EDUs*	*# Relations*
undergrad 1	27	1342	1216
undergrad 2	31	1378	1302
undergrad 3	31	1376	1173
grad 1	47	1387	1390
grad 2	48	1314	1321

Table 1: Statistics on documents, EDUs and Relations.

4 Experiments and Results

Using ANNODIS's annotated corpus, we checked for all EDUs π, whether π was attached to a constituent in the SDRS built from the previous EDUs in a way that violated the RFC. Given a discourse as a series of EDUs $\pi_1, \pi_2, \ldots, \pi_n$, we constructed for each π_i the corresponding sub-graph and calculated the set of nodes on the RF of this subgraph. We then checked whether the EDU π_{i+1} was attached to a node that was found in this set. We also checked whether any newly created complex segment was attached to a node on the RF of this sub-graph.

4.1 Calculating the Nodes at the RF

To calculate the nodes on the RF, we slightly extended the annotated graphs, in order to add im-

[6] Frame adverbials are sentence initial adverbial phrases that can either be temporal, spatial or "topical" (*in Chemistry*).

plied relations left out by the annotators.[7]

Disconnected Graphs While checking the RFC for the attachment of a node n, the SDRS graph at this point might consist of 2 or more disjoint subgraphs which get connected together at a later point. Because we did not want to decide which way these graphs should be connected, we defined a right frontier for each one using its own LAST. We then calculated the RF for each one of them and set the set of available nodes to be those in the union of the RFs of the disjoint subgraphs. If the subgraphs were not connected at the end of the incremental process in a way that conformed to RFC, we counted this as a violation. Annotators did not always provide us with a connected graph.

Postponed Decisions SDRT allows for the attachment not only of EDUs but also of subgraphs to an available node in the contextually given SDRS. For instance, in the following example, the intended meaning is given by the graph in which the Contrast is between the first label and the complex constituent composed of the disjunction of π_2 and π_3.

(π_1) Bill doesn't like sports. (π_2) But Sam does. (π_3) Or John does.

$$\begin{tikzcd} \pi_1 \arrow[rr, "Contrast"'] & & \pi' \arrow[dl, dash] \arrow[dr, dash] & \\ & \pi_2 \arrow[rr, "Altern."'] & & \pi_3 \end{tikzcd}$$

Naive annotators attached subgraphs instead of EDUs to the RF with some regularity (around 2%). This means that an EDU π_{i+1} could be attached to a node that was not present in the subgraph produced by $\pi_1, \ldots, \pi_i$. There were two main reasons for this: (1) π_{i+1} came from a syntactically fronted clause, a parenthetical or apposition in a sentence whose main clause produced π_{i+2} and π_{i+1} was attached to π_{i+2}; (2) π_{i+1} was attached to a complex segment $[\ldots, \pi_{i+1}, \ldots, \pi_{i+k}, \ldots]$ which was not yet introduced in the subgraph.

Since the nodes to which π_{i+1} is attached in such cases are not present in the graph, *by definition* they are not in the RF and they could be counted as violations. Nonetheless, if the nodes which connect nodes like π_{i+1} eventually link up to the incrementally built SDRS in the right way, π_{i+1} might eventually end up linked to something on the RF. For this reason, we postponed the decision on nodes like π_{i+1} until the nodes to which they are attached were explicitly introduced in the SDRS.

The Coherence of Complex Segments In an SDRS, several EDUs may combine to form a complex segment α that serves as a term for a discourse relation R. The interpretation of the SDRS implies that all of α's constituents contribute to the rhetorical function specified by R. This implies that the coordinating relation *Continuation* holds between the EDUs inside α, unless there is some other relation between them that is incompatible with Continuation (like a subordinating relation). Continuations are often used in SDRT (Asher, 1993; Asher and Lascarides, 2003). During the annotation procedure, our subjects did not always explicitly link the EDUs within a complex segment. In order to enforce the coherence of those complex segments we added *Continuation* relations between the constituents of a complex segment *unless* there was already another path between those constituents.

Expanding Continuations Consider the following discourse:

(4) [John, [who owns a chain of restaurants]$_{\pi_2}$, [and is a director of a local charity organization,]$_{\pi_3}$ wanted to sell his yacht.]$_{\pi_1}$ [He couldn't afford it anymore.]$_{\pi_4}$

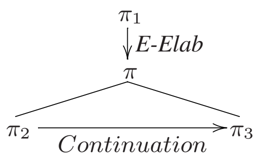
Annotators sometimes produced the following SDRT graph for the first three EDUs of this discourse:

(5)
$$\begin{tikzcd} \pi_1 \arrow[d, "E\text{-}Elab"] & \\ \pi_2 \arrow[r, "Continuation"'] & \pi_3 \end{tikzcd}$$

In this case the only open node is π_3 due to the coordinating relation *Continuation*. Nonetheless, π_4 should be attached to π_1, without violating the RFC. Indeed, SDRT's definition of the Continuation relation enforces that if we have $R(\pi_1, \pi_2)$ and Continuation(π_2, π_3) then we actually have the complex segment $[\pi_2, \pi_3]$ with $R(\pi_1, [\pi_2, \pi_3])$. So there is in fact a missing complex segment in (5). The proper SDRS graph of (4) is:

[7] In similar work on TimeML annotations, Setzer et al. (2003; Muller and Raymonet (2005) add implied relations to annotated, temporal graphs.

(6)



which makes π_1 an available attachment site for π_4. Such implied constituents have been added to the SDRS graphs.

Factoring Related to the operation of Expansion, SDRT's definition of Continuation and various subordinating relations also requires that if we have $R(a, [\pi_1, \pi_2, \ldots, \pi_n])$ where $[\pi_1, \pi_2, \ldots, \pi_n]$ is a complex segment with $\pi_1, \ldots \pi_n$ linked by Continuation and R is Elaboration, Entity-Elaboration, Frame, Attribution, or Commentary, then we also have $R(a, \pi_i)$ for each i. We added these relations when they were missing.

4.2 Results

With the operations just described, we added several inferred relations to the graph. We then calculated statistics concerning the percentage of attachments for which the RFC is respected using the following formula:

$$RFC_{\text{EDU}} = \frac{\texttt{\# EDUs attached to the RF}}{\texttt{\# EDUs in total}}$$

As we explained, an EDU can be attached to an SDRT graph directly by itself or indirectly as part of a bigger complex segment. In order to calculate the nominator we determine first whether an EDU directly attaches to the graph's RF, and if that fails we determine whether it is part of a larger complex segment which is attached to the graph's RF. The results obtained are shown in the first two columns of table 2. The RFC is respected by at least some attachment decision 95% of the time—i.e., 95% of the EDUs get attached to another node that is found on the RF. The breakdown across our annotators is given in table 2.

SDRT allows for multiple attachments of an EDU to various nodes in an SDRS; *e.g.* while an EDU may be attached via one relation to a node on the RF, it may be attached to another node off the RF. To take account of all the attachments for a given EDU, we need another way of measuring the percentage of attachments that respects the RFC. So we counted the ways each EDU is related to a node in the SDRS for the previous text and then divided the number of attachment decisions that respect the RFC by the total number of attachment decisions—*i.e.* :

$$RFC_r = \frac{\texttt{\# RF attachment decisions}}{\texttt{\# Total attachment decisions}}$$

.

annotator	RFC_{EDU}	RFC_r
undergrad 1	98.57%	91.28%
undergrad 2	98.12%	94.39%
undergrad 3	91.93%	89.17%
grad 1	94.38%	86.54%
grad 2	92.68%	83.57%
Mean for all annotators	95.24%	88.91%
Mean for 3 undergrad	96.17%	91.71%

Table 2: The % with which each annotator has respected SDRT's RFC using the EDU and attachment decision measures.

The third column of table 2 shows that having a stable annotation manual and GLOZZ improved the results across our two annotator populations, even though the annotation manual did not say anything about RFC or about the structure of the discourse graphs. Moreover, the distribution of violations of the RFC follows a power law and only 4.56% of the documents contained more than 5 violations. This is strong evidence that there is little propagation of violations.

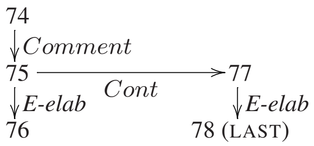
5 Analysis of Presumed Violations

Although 95% of EDUs attach to nodes on the RF of an SDRT graph, 5% of EDUs don't. SDRT experts performed a qualitative analysis of some of these presumed violations. In many cases, the experts judged that the presumed violations were due to click-errors: sometimes the annotators simply clicked on something that did not translate into a segment. Sometimes, the experts judged that the annotators picked the wrong segment to attach a new segment or the wrong type of relation during the construction of the SDRT graph. For example, in the graph that follows the relation between segments 74 and 75 is not a *Comment* but an *Entity-Elaboration*.

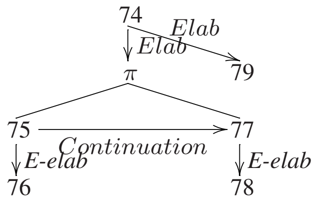
As expected, there were also *"structural"* errors, arising from a lack or a misuse of complex segments. Here is a typical example (translated from the original French):

> [Around her,]_74 [we should mention Joseph Racaille]_75 [responsible for the magnificent arrangements,]_76 [Christophe Dupouy]_77 [regular associate of Jean-Louis Murat responsible for mixing,]_78 [without forgetting her two guardian angels:]_79 [her agent Olivier Gluzman]_80 [who signed after a love at first sight,]_81 [and her husband Mokhtar]_82 [who has taken care of the family]_83

Here is the annotated structure up to EDU 78:

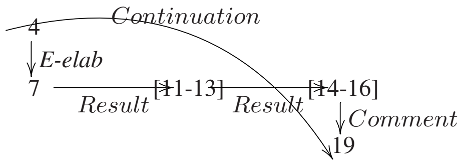


Note that the attachment of 77 to 75 is non-local and non-adjacent. The annotator then attaches EDU 79 to 75 which is blocked from the RF due to the $Continuation$ coordinating relation. By not having created a complex segment due the enumeration that includes EDUs 75 to 78, the annotator had no option but to violate the RF. Here is the proper SDRT graph for segments 74 to 79 (where the attachment of 79 to 74 is also both non-local and non-adjacent):



In this case, before the introduction of EDU 79, EDU 78 is LAST and by consequence 77, π and 74 are on the RF. Attaching 79 to 74 is thus legitimate.

We also found more interesting examples of right frontier violations. One annotator produced a graph for a story which is about the attacks of 9/11/2001 and is too long to quote here. A simplified graph of the first part of the story is shown below. EDU 4 elaborates on the main event of the story but it is not on the RF for 19. However, 19 is the first recurrence of the complex definite description *le 11 septembre 2001* since the title and the term's definition in EDU 4.



This reuse of the full definite description could be considered a case of SDRT's discourse subordination.

6 RFC and distances of attachment

Our empirical study vindicates SDRT's RFC, but it also has computational implications. Using the RFC dramatically diminishes the number of attachment possibilities and thus greatly reduces the search space for any incremental discourse parsing algorithm.[8] The mean of nodes that are open on the RF at any given moment on our ANNODIS data is 16.43% of all the nodes in the graph.

Our data also allowed us to calculate the distance of attachment sites from LAST, which could be an important constraint on machine learning algorithms for constructing discourse structures. Given a pair of constituents (π_i, π_j) distance is calculated either *textually* (the number of intervening EDUs between π_i and π_j) or *topologically* (the length the shortest path between π_i and π_j). Topological distance, however, does not take into account the fact that a textually further segment is cognitively less salient. Moreover, this measure can give the same distance to nodes that are textually far away between them due to long distance pop-ups (Asher and Lascarides, 2003). A purely textual distance, on the other hand, gives the same distance to an EDU π_i and a complex segment $[\pi_1, \ldots, \pi_i]$ even if π_1 and π_i are textually distant (since both have the same span end). We used a measure combining both. The distance scheme that we used assigns to each EDU its textual distance from LAST in the graph under consideration, while a complex segment of rank 1 gets a distance which is computed from the highest distance of their constituent EDUs plus 1. For a constituent σ of rank n we have:

$$Dist = Max\{\text{dist}(x) \colon x \text{ in } \sigma\} + n$$

[8] An analogous approach for search space reduction is followed by duVerle and Prendinger (2009) who use the "Principle of Sequentiality" (Marcu, 2000), though they do not say how much the search space is reduced.

The distribution of attachment follows a power law with 40% of attachments performed non-locally, that is on segments of distance 2 or more (figure 1). This implies that the distance between candidate attachment sites that are on the RF is an important feature for an ML algorithm. It is important to note at this point that following the baseline approach of always attaching on the LAST misses 40% of attachments. We also have 20.38% of the non-local, non-adjacent attachments in our annotations. So an RST parser using Marcu's (2000) adjacency constraint as do duVerle and Prendinger (2009) would miss these.

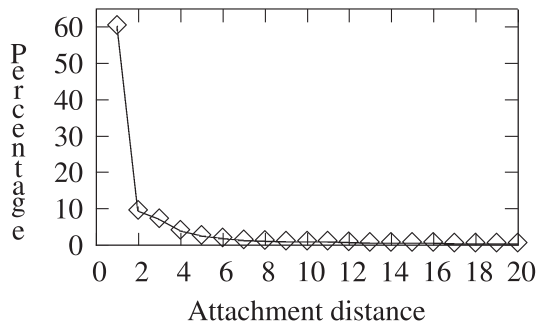


Figure 1: Distribution of attachment distance

7 Related Work

Several studies have shown that the RFC may be violated as an anaphoric constraint when there are other clues, content or linguistic features, that determine the antecedent. (Poesio and di Eugenio, 2001; Holler and Irmen, 2007; Asher, 2008; Prévot and Vieu, 2008), for example, show that anaphors such as definite descriptions and complex demonstratives, which often provide enough content on their own to isolate their antecedents, or pronouns in languages like German which must obey gender agreement, might remain felicitous although the discourse relations between them and their antecedents might violate the RFC. Usually there are few linguistic clues that help find the appropriate antecedent to a discourse relation, in contrast to the anaphoric expressions mentioned above. Exceptions involve stylistic devices like direct quotation that license discourse subordination. Thus, SDRT predicts that RFC violations for discourse attachments should be much more rare than those for the resolution of anaphors that provide linguistic clues about their antecedents.

As regards other empirical validation of various versions of the RFC for the attachment of discourse constituents, Wolf and Gibson (2006) show an RST-like RFC is not supported in their corpus GraphBank. Our study concurs in that some 20% of the attachments in our corpus cannot be formulated in RST.[9] On the other hand, we note that because of the 2 dimensional nature of SDRT graphs and because of the caveats introduced by structural relations and discourse subordination, the counterexamples from GraphBank against, say, RST representations do not carry over straightforwardly to SDRSs. In fact, once these factors are taken into account, the RFC violations in our corpus and in GraphBank are roughly about the same.

8 Conclusions

We have shown that SDRT's RFC has strong empirical support: the attachments of our 3 completely naive annotators fully comply with RFC 91.7% of the time and partially comply with it 96% of the time. As a constraint on discourse parsing SDRT's RFC, we have argued, is both empirically and computationally motivated. We have also shown that non-local attachments occur about 40% of the time, which implies that attaching directly on the LAST will not yield good results. Further, many of the non local attachments do not respect RST's adjacency constraint. We need SDRT's RFC to get the right attachment points for our corpus. We believe that empirical studies of the kind we have given here are essential to finding robust and useful features that will vastly improve discourse parsers.

[9]One other study we are aware of is Sassen and Kühnlein (2005), who show that in chat conversations, the RFC does not always hold unconditionally. Since this genre of discourse is not always coherent, it is expected that the RFC will not always hold here.

References

Asher, N. and A. Lascarides. 2003. *Logics of Conversation*. Studies in Natural Language Processing. Cambridge University Press, Cambridge, UK.

Asher, N. and L. Vieu. 2005. Subordinating and coordinating discourse relations. *Lingua*, 115(4):591–610.

Asher, N. 1993. *Reference to Abstract Objects in Discourse*. Kluwer Academic Publishers.

Asher, N. 2007. A large view of semantic content. *Pragmatics and Cognition*, 15(1):17–39.

Asher, N. 2008. Troubles on the right frontier. In Benz, A. and P. Kühnlein, editors, *Constraints in Discourse*, Pragmatics and Beyond New Series, chapter 2, pages 29–52. John Benjamins Publishing Company.

Baldridge, J., N. Asher, and J. Hunter. 2007. Annotation for and robust parsing of discourse structure on unrestricted texts. *Zeitschrift fur Sprachwissenschaft*, 26:213–239.

Danlos, L. 2008. Strong generative capacity of rst, sdrt and discourse dependency dags. In Benz, A. and P. Kühnlein, editors, *Constraints in Discourse*, Pragmatics and Beyond New Series, pages 69–95. John Benjamins Publishing Company.

duVerle, D. and H. Prendinger. 2009. A novel discourse parser based on support vector machine classification. In *Proceedings of ACL*, pages 665–673, Suntec, Singapore, August.

Hardt, D. and M. Romero. 2004. Ellipsis and the structure of discourse. *Journal of Semantics*, 21:375–414, November.

Hardt, D., N. Asher, and J. Busquets. 2001. Discourse parallelism, scope and ellipsis. *Journal of Semantics*, 18:1–16.

Holler, A. and L. Irmen. 2007. Empirically assessing effects of the right frontier constraint. In *Anaphora: Analysis, Algorithms and Applications*, pages 15–27. Springer, Berlin/Heidelberg.

Lascarides, A. and N. Asher. 1993. Temporal interpretation, discourse relations and commonsense entailment. *Linguistics and Philosophy*, 16(5):437–493.

Lee, A., R. Prasad, A. Joshi, and B. Webber. 2008. Departures from tree structures in discourse: Shared arguments in the penn discourse treebank. In *Constraints in Discourse (CID '08)*, pages 61–68.

Mann, W. and S. Thompson. 1987. Rhetorical structure theory: A framework for the analysis of texts. Technical Report ISI/RS-87-185, Information Sciences Institute, Marina del Rey, California.

Mann, W. and S. Thompson. 1988. Rhetorical structure theory: Towards a functional theory of text organization. *Text*, 8(3):243–281.

Marcu, D. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. The MIT Press.

Muller, P. and A. Raymonet. 2005. Using inference for evaluating models of temporal discourse. In *12th International Symposium on Temporal Representation and Reasoning*, pages 11–19. IEEE Computer Society Press.

Poesio, M. and B. di Eugenio. 2001. Discourse structure and anaphoric accessibility. In *Proc. of the ESSLLI Workshop on Discourse Structure and Information Structure*, August.

Polanyi, L. 1985. A theory of discourse structure and discourse coherence. In Kroeber, P. D., W. H. Eilfort, and K. L. Peterson, editors, *Papers from the General Session at the 21st Regional Meeting of the Chicago Linguistics Society*.

Polanyi, L. 1988. A formal model of the structure of discourse. *Journal of Pragmatics*, 12:601–638.

Prévot, L. and L. Vieu. 2008. The moving right frontier. In Benz, A. and P. Kühnlein, editors, *Constraints in Discourse*, Pragmatics and Beyond New Series, chapter 3, pages 53–66. John Benjamins Publishing Company.

Sassen, C. and P. Kühnlein. 2005. The right frontier constraint as conditional. In *Computational Linguistics and Intelligent Text Processing*, Lecture Notes in Computer Science (LNCS), pages 222–225.

Setzer, A., R. Gaizauskas, and M. Hepple. 2003. Using semantic inferences for temporal annotation comparison. In *Proceedings of the Fourth International Workshop on Inference in Computational Semantics (ICoS-4)*.

Webber, B. 1988. Title discourse deixis and discourse processing. Technical Report MS-CIS-88-75, University of Pennsylvania, Department of Computer and Information Science, September.

Wolf, F. and E. Gibson. 2006. *Coherence in Natural Language: Data Stuctures and Applications*. The MIT Press.

Identifying Multi-word Expressions by Leveraging Morphological and Syntactic Idiosyncrasy

Hassan Al-Haj
Language Technologies Institute
Carnegie Mellon University
hhaj@cs.cmu.edu

Shuly Wintner
Department of Computer Science
University of Haifa
shuly@cs.haifa.ac.il

Abstract

Multi-word expressions constitute a significant portion of the lexicon of every natural language, and handling them correctly is mandatory for various NLP applications. Yet such entities are notoriously hard to define, and are consequently missing from standard lexicons and dictionaries. Multi-word expressions exhibit idiosyncratic behavior on various levels: orthographic, morphological, syntactic and semantic. In this work we take advantage of the morphological and syntactic idiosyncrasy of Hebrew noun compounds and employ it to extract such expressions from text corpora. We show that relying on linguistic information dramatically improves the accuracy of compound extraction, reducing over one third of the errors compared with the best baseline.

1 Introduction

Multi-word expressions (MWEs) are notoriously hard to define. They span a range of constructions, from completely frozen, semantically opaque idiomatic expressions, to frequent but morphologically productive and semantically compositional collocations. Various linguistic processes (orthographic, morphological, syntactic, semantic, and cognitive) apply to MWEs in idiosyncratic ways. Notably, MWEs blur the distinction between the lexicon and the grammar, since they often have some properties of words and some of phrases.

In this work we define MWEs as expressions whose linguistic properties (morphological, syntactic or semantic) are not directly derived from the properties of their word constituents. This is a functional definition, driven by a practical motivation: any natural language processing (NLP) application that cares about morphology, syntax or semantics must consequently store MWEs in the lexicon.

MWEs are numerous and constitute a significant portion of the lexicon of any natural language. They are a heterogeneous class of constructions with diverse sets of characteristics. Morphologically, some MWEs allow some of their constituents to freely inflect while restricting (or even preventing) the inflection of other constituents. MWEs may allow constituents to undergo non-standard morphological inflections that they would not undergo in isolation. Some MWEs contain words that never occur outside the context of the MWE. Syntactically, some MWEs appear in one rigid pattern (and a fixed order), while others permit various syntactic transformations. Semantically, the compositionality of MWEs (i.e., the degree to which the meaning of the whole expression results from combining the meanings of its individual words when they occur in isolation) is gradual.

These morphological, syntactic and semantic idiosyncrasies make MWEs a challenge for NLP applications (Sag et al., 2002). They are even more challenging in languages with complex morphology, because of the unique interaction of morphological and orthographic processes with the lexical specification of MWEs (Oflazer et al., 2004; Alegria et al., 2004).

Because the idiosyncratic features of MWEs cannot be predicted on the basis of their component words, they must be stored in the lexicon of NLP applications. Handling MWEs correctly is beneficial for a variety of applications, including information retrieval, building ontologies, text alignment, and machine translation. Automatic identification and corpus-based extraction of MWEs is thus crucial for such (and several other) applications.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 10–18,
Beijing, August 2010

In this work we describe an approach that leverages the morphological and syntactic idiosyncrasy of a certain class of Hebrew[1] MWEs, namely noun compounds, to help identify such expressions in texts. While the main contribution of this work is a system that can distinguish between MWE and non-MWE instances of a particular construction in Hebrew, thereby facilitating faster and more accurate integration of MWEs in a large-coverage lexicon of the language, we believe that it carries added value to anyone interested in MWEs. The technique that we propose here should be applicable in principle to any language in which MWEs exhibit linguistically idiosyncratic behavior.

We describe the properties of Hebrew noun-noun constructions in Section 2, and specify the irregularities exhibited by compounds. Section 3 presents the experimental setup and the main results. Compared with the best (collocation-based) baseline, our approach reduces over 30% of the errors, yielding accuracy of over 80%. We discuss related work in Section 4 and conclude with suggestions for future research.

2 Hebrew noun-noun constructions

We focus on Hebrew noun-noun constructions; these are extremely frequent constructions, and while many of them are fully compositional, others, called *noun compounds* (or just *compounds*) here, are clearly MWEs. We first discuss the general construction and then describe the peculiar, idiosyncratic properties of compounds.

2.1 The general case

Hebrew nouns inflect for number (singular and plural) and, when the noun denotes an animate entity, for gender (masculine and feminine). In addition, nouns come in three *states*: indefinite, definite and a *construct* state that is used in genitive constructions. Table 1 demonstrates the paradigm.

A noun-noun construction (henceforth NNC) consists of a construct-state noun, called *head* here, followed by a noun phrase, the *modifier* (Borer, 1988; Borer, 1996; Glinert, 1989).

[1]To facilitate readability we use a transliteration of Hebrew using Roman characters; the letters used, in Hebrew lexicographic order, are *abgdhwzxTiklmns‘pcqršt*.

State	M/Sg	F/Sg	M/Pl	F/Pl
indefinite	*ild*	*ildh*	*ildim*	*ildwt*
definite	*hild*	*hildh*	*hildim*	*hildwt*
construct	*ild*	*ildt*	*ildi*	*ildwt*

Table 1: The noun paradigm, demonstrated on *ild* “child”

The semantic relation between the two is usually, but not always, related to possession (Levi, 1976). Construct-state nouns only occur in the context of NNC, and can never occur in isolation. When a NNC is definite, the definite article is expressed on its modifier (Wintner, 2000).

In the examples below, we explicitly indicate construct-state nouns by the morpheme ‘.CONST’ in the gloss; and definite nouns are indicated by the morpheme ‘the-’. We provide both a literal and a non-literal meaning of the MWE examples. Expressions that have a literal, but not the expected MWE meaning, are preceded by ‘#’.

Example 1 (Noun-noun constructions)

hxlTt *hw‘dh*
*decision.*CONST *the-committee*
“the committee decision”

‘wrk *h‘itwn*
*editor.*CONST *the-journal*
“the journal editor”

‘wrk *din*
*editor.*CONST *law*
“law editor” ⟹ lawyer

bti *xwlim*
*houses.*CONST *patients*
“patient houses” ⟹ hospitals

2.2 Noun compounds: Linguistic properties

While many of the NNCs are free, compositional combinations of words, some are not; we use the term *noun compounds* for the latter group. Compounds typically (but not necessarily) have non-compositional meaning; presumably due to their opaque, more lexical meaning, they also differ from other NNCs in their morphological and syntactic behavior. Some of these distinctive properties are listed below, to motivate the methodology that we propose in Section 3 to distinguish between compounds and non-MWE NNCs.

2.2.1 Limited inflection

When a NNC consists of two nouns, the second can typically occur in either singular or plural form. Compounds often limit the possibilities to only one of those.

Example 2 (No plural form of the modifier)

‘wrki	*h‘itwnim*
*editors-.*CONST	*the-journals*
“the journals’ editors”	

‘wrki	*hdin*
*editors.*CONST	*the-law*
“the law editors” ⟹ the lawyers	

#wrki	*hdinim*
*editors.*CONST	*the-laws*

Example 3 (No singular form of the modifier)

kiwwn	*hrwx*
*direction.*CONST	*the-wind*
“the wind’s direction”	

kiwwn	*hrwxwt*
*direction.*CONST	*the-winds*
“the winds’ direction”	

šwšnt	*h-rwxwt*
*lily.*CONST	*the-winds*
“lily of the winds” ⟹ compass rose	

#šwšnt	*h-rwx*
*lily.*CONST	*the-wind*

2.2.2 Limited syntactic variation

Since NNCs typically denote genitive (possessive) constructions, they can be paraphrased by a construction that uses the genitive preposition *šl* “of” (or, in some cases, other prepositions). These syntactic variants are often restricted in the case of compounds.

Example 4 (Limited paraphrasing)

h‘wrk	*šl*	*h‘itwn*
the-editor	*of*	*the-journal*
“the journal editor”		

#h‘wrk	*šl*	*hdin*
the-editor	*of*	*the-law*

Example 5 (Limited paraphrasing)

m‘il	*cmr*
*coat.*CONST	*wool*
“wool coat”	

m‘il	*mcmr*
coat	*from-wool*
“wool coat”	

cmr	*pldh*
*wool.*CONST	*steel*
“steel wool” ⟹ steel wool	

#cmr	*mpldh*
wool	*from-steel*

2.2.3 Limited syntactic modification

NNCs typically allow adjectival modification of either of their constituents. Since compounds tend to be more semantically opaque, it is often only possible to modify the entire compound, but not any of the constituents. In the following example, note that *‘wrkt* “editor” is feminine, whereas *‘itwn* “journal” is masculine; adjectives must agree on gender with the noun they modify.

Example 6 (Limited adjectival modification)

’wrkt	*h’itwn*
*editor-f.*CONST	*the-journal-m*
“the journal editor”	

‘wrkt	*h‘itwn*	*hxdšh*
*editor-f.*CONST	*the-journal-m*	*the-new-f*
“the new editor of the journal”		

‘wrkt	*h‘itwn*	*hxdš*
*editor-f.*CONST	*the-journal-m*	*the-new-m*
“the editor of the new journal”		

‘wrkt	*hdin*	*hxdšh*
*editor-f.*CONST	*the-law-m*	*the-new-f*
“the new law editor” ⟹ the new lawyer		

#‘wrkt	*hdin*	*hxdš*
*editor-f.*CONST	*the-law-m*	*the-new-m*

2.2.4 Limited coordination

Two NNCs that share a common head can be conjoined using the coordinating conjunction *w* “and”. This possibility is often blocked in the case of compounds.

Example 7 (Limited coordination)

mwsdwt	*xinwk*	*wbriawt*
*institutions.*CONST	*education*	*and-health*
“education and health institutions”		

bti	*spr*
*houses.*CONST	*book*
“book houses” ⟹ schools	

bti	*xwlim*	
*houses.*CONST	*patients*	
"patient houses" $\Longrightarrow$ *hospitals*		
#bti	*spr*	*wxwlim*
*houses.*CONST	*book*	*and-patients*

3 Identification of noun compounds

In this section we describe a system that identifies noun compounds in Hebrew text, and extracts them in order to extend the lexicon. We capitalize on the morphological and syntactic irregularities of noun compounds described in Section 2.2.

Given a large monolingual corpus, the text is first morphologically analyzed and disambiguated. Then, all NNCs (candidate noun compounds) are extracted from the morphologically disambiguated text. For each candidate noun compound we define a set of features (Section 3.3) based on the idiosyncratic morphological and syntactic properties defined in Section 2.2. These features inform a support vector machine classifier which is then used to identify the noun compounds in the set of NNCs with high accuracy (Section 3.5).

3.1 Resources

We use (a subset of) the Corpus of Contemporary Hebrew (Itai and Wintner, 2008) which consists of four sub-corpora: The *Knesset* corpus contains the Israeli parliament proceedings from 2004-2005; the *Haaretz* corpus contains articles from the Haaretz newspaper from 1991; *TheMarker* corpus contains financial articles from the TheMarker newspaper from 2002; and the *Arutz 7* corpus contains newswire articles from 2001-2006. Corpora sizes are listed in Table 2.

Corpus	Number of tokens
Knesset	12,742,879
Harretz	463,085
The Marker	684,801
Arutz 7	7,714,309
Total	21,605,074

Table 2: Corpus data

The entire corpus was morphologically analyzed (Yona and Wintner, 2008; Itai and Wintner, 2008) and POS-tagged (Bar-haim et al., 2008); note that no syntactic parser is available for Hebrew. From the morphologically disambiguated corpus, we extract all bi-grams in which the first token is a noun in the construct state and the second token is a noun that is not in the construct state, i.e., all two-word NNC *candidates*.

3.2 Annotation

For training and evaluation, we select the NNCs that occur at least 100 times in the corpus, yielding 1060 NNCs. These NNCs were annotated by three annotators, who were asked to classify them to the following four groups: compounds (+); non-compounds (–); unsure (0); and errors of the morphological processor (i.e., the candidate is not a NNC at all). Table 3 lists the number of candidates in each class.

Annotator	+	–	0	err
1	314	332	238	176
2	335	403	179	143
3	400	630	16	14

Table 3: NNC classification by annotator

We adopt a conservative approach in combining the three annotations. First, we eliminate 204 NNCs that were tagged as errors by at least one annotator. For the remaining NNCs, a candidate is considered a compound or a non-compound only if all three annotators agree on its classification. This reduces the annotated data to 463 instances, of which 205 are compounds and 258 are clear cases of non-compound NNCs.[2]

3.3 Linguistically-motivated features

We define a set of features based on the idiosyncratic properties of noun compounds defined in Section 2.2. For each candidate NNC, we compute counts which reflect the likelihood of it exhibiting one of the linguistic properties.

Refer back to Section 2.2. We focus on the property of limited inflection (Section 2.2.1), and define features 1–8 to reflect it. To reflect limited syntactic variation (Section 2.2.2) we define features 9–10. Feature 11 addresses the phenomenon

[2]This annotated corpus is freely available for download.

of limited coordination (Section 2.2.4). To reflect limited syntactic modification (Section 2.2.3) we define feature 12. .

For each NNC candidate N_1 N_2, the following features are defined:

1. The number of occurrences of the NNC in which both constituents are in singular.
2. The number of occurrences of the NNC in which N_1 is in singular and N_2 is in plural.
3. The number of occurrences of the NNC in which N_1 is in plural and N_2 is in singular.
4. The number of occurrences of the NNC in which both constituents are in plural.
5. The number of occurrences of N_1 in plural outside the expression.
6. The number of occurrences of N_1 in singular outside the expression.
7. The number of occurrences of N_2 in plural outside the expression.
8. The number of occurrences of N_2 in singular outside the expression.
9. The number of occurrences of N_1 *šl* N_2 "N_1 of N_2" in the corpus.
10. The number of occurrences of N_1 *m* N_2 "N_1 from N_2" in the corpus.
11. The number of occurrences of N_1 N_2 *w* N_3 "N_1 N_2 and N_3" in the corpus, where N_3 is an indefinite, non-construct-state noun.
12. The number of occurrences of N_1 N_2 *Adj* in the corpus, where the adjective *Adj* agrees with N_2 on both gender and number, while disagreeing with N_1 on at least one of these attributes.

We also define four features that represent known collocation measures (Evert and Krenn, 2001): Point-wise mutual information (PMI); T-Score; log-likelihood; and the raw frequency of N_1 N_2 in the corpus.[3]

[3]A detailed description of these measures is given by Manning and Schütze (1999, Chapter 5); see also http://www.collocations.de/, where several other association measures are discussed as well.

3.4 Training and evaluation

For each NNC in the annotated set of Section 3.2 we create a vector of the 16 features described in Section 3.3 (12 linguistically-motivated features plus four collocation measures). We obtain a list of 463 instances, of which 205 are positive examples (noun compounds) and 258 are negative. We use this set for training and evaluation of a two class soft margin SVM classifier (Chang and Lin, 2001) with a radial basis function kernel. We experiment below with different combinations of features, where for each combination we use 10-fold cross-validation over the 463 NNcs to evaluate the classifier. We report Precision, Recall, F-score and Accuracy (averaged over the 10 folds).

3.5 Results

The results of the different classifiers that we trained are given in Table 4. The first four rows of the table show the performance of classifiers trained using each of the four different collocation measure features alone. Both PMI and Log-likelihood outperform the other collocation measures, with an F-score of 60, which we consider our baseline. We also report the performance of two combinations of collocation measures, which yield small improvement. The best combinations provide accuracy of about 70% and F-score of 63.

The remaining rows report results using the linguistically-motivated features (LMF) of Section 3.3. These features alone yield accuracy of 77.75% and an F-score of 76. Adding also Log-likelihood improves F-score by 1.16 and accuracy by 1.29%. Finally, using Log-likelihood with a subset of the LMF consisting of features 1-2, 4-6, 9-10 and 12 (see below) yields the best results, namely accuracy of over 80% and F-score of 78.85, reflecting a reduction of over one third in classification error rate compared with the baseline.

3.6 Optimizing feature combination

We search for the combination of linguistically-motivated features that would yield the best performance. Training a classifier on all possible feature combinations is clearly infeasible. Instead, we follow a more efficient greedy approach, whereby we start with the best collocation mea-

Features	Accuracy	Precision	Recall	F-score
PMI	67.17	64.97	**56.09**	**60.20**
Frequency	60.47	60.00	32.19	41.90
T-Score	61.98	59.86	42.92	50.00
Log-likelihood	**69.33**	**71.42**	51.21	59.65
T-score+Log-likelihood	**70.62**	**71.42**	56.09	62.84
PMI+Log-likelihood	69.97	68.96	**58.53**	**63.32**
LMF	77.75	71.98	81.46	76.43
LMF+PMI	77.32	71.18	**81.95**	76.19
LMF+Log-likelihood	79.04	73.68	**81.95**	77.59
Log-likelihood+LMF[1-2,4-6,9-10,12]	**80.77**	**76.85**	80.97	**78.85**

Table 4: Results: 10-Fold accuracy, precision, recall, and F-score for classifiers trained using different combinations of features. *LMF* stands for linguistically-motivated features

sure, Log-likelihood, and add other features one at a time, in the order in which they are listed in Section 3.3. After adding each feature the classifier is retrained; the feature is retained in the feature set only if adding it improves the 10-fold F-score of the current feature set.

Table 5 lists the results of this experiment. For each feature set the difference in the 10-fold F-score compared to the previous feature set is listed in parentheses. The results show that the best feature combination improves the F-score by 1.26, compared with using all features. This experiments shows that features 3, 7, 8 and 11 turn out not to be useful, and the classifier is more accurate without them. We also tried this approach with PMI as the starting feature, with very similar results.

Feature set	F-score	
Log-likelihood	59.65	
Log-likelihood,1	60.34	(+0.68)
Log-likelihood,1-2	65.42	(+5.08)
Log-likelihood,1-3	64.87	(-0.54)
Log-likelihood,1-2,4	66.66	(+1.78)
Log-likelihood,1-2,4-5	70.00	(+3.33)
Log-likelihood,1-2,4-6	74.37	(+4.37)
Log-likelihood,1-2,4-7	73.78	(–0.58)
Log-likelihood,1-2,4-6,8	73.58	(–0.79)
Log-likelihood,1-2,4-6,9	78.72	(+4.35)
Log-likelihood,1-2,4-6,9-10	78.83	(+0.10)
Log-likelihood,1-2,4-6,9-11	77.37	(–1.46)
Log-likelihood,1-2,4-6,9-10,12	78.85	(+0.02)

Table 5: Optimizing the set of linguistically-motivated features

4 Related work

There has been a growing awareness in the research community of the problems that MWEs pose, both in linguistics and in NLP (Villavicencio et al., 2005). Recent works address the definition, lexical representation and computational processing of MWEs, as well as algorithms for extracting them from data.

Focusing on acquisition of MWEs, early approaches concentrated on their collocational behavior (Church and Hanks, 1989). Pecina (2008) compares 55 different association measures in ranking German Adj-N and PP-Verb collocation candidates. This work shows that combining different collocation measures using standard statistical-classification methods (such as Linear Logistic Regression and Neural Networks) gives a significant improvement over using a single collocation measure. Our results show that this is indeed the case, but the contribution of collocation methods is limited, and more information is needed in order to distinguish frequent collocations from bona fide MWEs.

Other works show that adding linguistic information to collocation measures can improve identification accuracy. Several approaches rely on the semantic opacity of MWEs; but very few semantic resources are available for Hebrew (the Hebrew WordNet (Ordan and Wintner, 2007), the only lexical semantic resource for this language, is small and too limited). Instead, we capitalize on the morphological and syntactic irregularities that MWEs exhibit, using computational resources that are more readily-available.

Ramisch et al. (2008) evaluate a number of association measures on the task of identifying English Verb-Particle Constructions and German Adjective-Noun pairs. They show that adding linguistic information (mostly POS and POS-sequence patterns) to the association measure yields a significant improvement in performance over using pure frequency. We follow this line of research by defining a number of syntactic patterns as a source of linguistic information. In addition, our linguistic features are much more specific to the phenomenon we are interested in, and the syntactic patterns are enriched by morphological information pertaining to the idiosyncrasy of MWEs; we believe that this explains the improved performance compared to the baseline.

Several works address the *lexical fixedness* or *syntactic fixedness* of (certain types of) MWEs in order to extract them from texts. An expression is considered lexically fixed if replacing any of its constituents by a semantically (and syntactically) similar word generally results in an invalid or literal expression. Syntactically fixed expressions prohibit (or restrict) syntactic variation.

For example, Van de Cruys and Villada Moirón (2007) use lexical fixedness to extract Dutch Verb-Noun idiomatic combinations (VNICs). Bannard (2007) uses syntactic fixedness to identify English VNICs. Another work uses both the syntactic and the lexical fixedness of VNICs in order to distinguish them from non-idiomatic ones, and eventually to extract them from corpora (Fazly and Stevenson, 2006). While these approaches are in line with ours, they require lexical semantic resources (e.g., a database that determines semantic similarity among words) and syntactic resources (parsers) that are unavailable for Hebrew (and many other languages). Our approach only requires morphological processing, which is more readily-available for several languages.

Another unique feature of our work is that it computationally addresses Hebrew (and, more generally, Semitic) MWEs for the first time. Berman and Ravid (1986) define the *dictionary degree* of noun compounds in Hebrew as their closeness to a single word from a grammatical point of view, as judged by the manner in which they are grasped by language speakers. A group of 120 Hebrew speakers were asked to assign a dictionary degree (from 1 to 5) to a list of 30 noun compounds. An analysis of the questionnaire results revealed that language speaker share a common dictionary, where the highest degree of agreement was achieved on the ends of the dictionary degree spectrum. Another conclusion is that both the pragmatic uses of the noun compound and the semantic relation between its constituents define the dictionary degree of the compound. Not having access to semantic and pragmatic knowledge, we are trying to approximate it using morphology.

Attia (2005) proposes methods to process fixed, semi-fixed, and syntactically-flexible *Arabic* MWEs (adopting the classification and the terminology of Sag et al. (2002)). Fabri (2009) provides an overview of the different types of compounds (14 in total) in present-day Maltese, focusing on one type of compounds consisting of an adjective followed by a noun. He also provides morphological, syntactic, and semantic properties of this group which distinguishes them from other non-compound constructions. Automatic identification of MWEs is not addressed in either of these works.

5 Conclusions and future work

We described a system that can identify Hebrew noun compounds with high accuracy, distinguishing them from non-idiomatic noun-noun constructions. The methodology we advocate is based on careful examination of the linguistic peculiarities of the construction, followed by corpus-based approximation of these properties via a general machine learning algorithm that is fed with features based on the linguistic properties. While our application is limited to a particular construction in a particular language, we are confident that it can be equally well applied to other constructions and other languages, as long as the targeted MWEs exhibit a consistent set of irregular features (especially in the morphology).

This work can be extended in various directions. Addressing other constructions is relatively easy, and requires only a theoretical linguistic investigation of the construction. We are currently interested in extending the system to cope also with Adjective-Noun, Noun-Adjective and Verb-Preposition constructions in Hebrew.

The accuracy of MWE acquisition systems can be further improved by combining our morphological and syntactic features with semantically informed features such as translational entropy computed from a parallel corpus (Villada Moirón and Tiedemann, 2006), or features that can capture the local linguistic context of the expression using latent semantic analysis (Katz and Giesbrecht, 2006). We are currently working on the former direction (Tsvetkov and Wintner, 2010b), utilizing a small Hebrew-English parallel corpus (Tsvetkov and Wintner, 2010a).

Finally, we are interested in evaluating the methodology proposed in this paper to other languages with complex morphology, in particular to Arabic. We leave this direction to future research.

Acknowledgments

This research was supported by THE ISRAEL SCIENCE FOUNDATION (grants No. 137/06, 1269/07). We are grateful to Alon Itai for his continuous help and advice throughout the course of this project, and to Bracha Nir for very useful comments. We also wish to thank Yulia Tsvetkov and Gily Chen for their annotation work.

References

Alegria, Iñaki, Olatz Ansa, Xabier Artola, Nerea Ezeiza, Koldo Gojenola, and Ruben Urizar. 2004. Representation and treatment of multiword expressions in Basque. In Tanaka, Takaaki, Aline Villavicencio, Francis Bond, and Anna Korhonen, editors, *Second ACL Workshop on Multiword Expressions: Integrating Processing*, pages 48–55, Barcelona, Spain, July. Association for Computational Linguistics.

Attia, Mohammed A. 2005. Accommodating multiword expressions in an lfg grammar. The ParGram Meeting, Japan September 2005, September. Mohammed A. Attia The University of Manchester School of Informatics mohammed.attia@postgrad.manchester.ac.uk.

Bannard, Colin. 2007. A measure of syntactic flexibility for automatically identifying multiword expressions in corpora. In *Proceedings of the Workshop on A Broader Perspective on Multiword Expressions*, pages 1–8. Association for Computational Linguistics.

Bar-haim, Roy, Khalil Sima'an, and Yoad Winter. 2008. Part-of-speech tagging of Modern Hebrew text. *Natural Language Engineering*, 14(2):223–251.

Berman, Ruth A. and Dorit Ravid. 1986. Lexicalization of noun compounds. *Hebrew Linguistics*, 24:5–22. In Hebrew.

Borer, Hagit. 1988. On the morphological parallelism between compounds and constructs. In Booij, Geert and Jaap van Marle, editors, *Yearbook of Morphology 1*, pages 45–65. Foris publications, Dordrecht, Holland.

Borer, Hagit. 1996. The construct in review. In Lecarme, Jacqueline, Jean Lowenstamm, and Ur Shlonsky, editors, *Studies in Afroasiatic Grammar*, pages 30–61. Holland Academic Graphics, The Hague.

Chang, Chih-Chung and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Church, Kenneth. W. and Patrick Hanks. 1989. Word association norms, mutual information and lexicography (rev). *Computational Linguistics*, 19(1):22–29.

Evert, Stefan and Brigitte Krenn. 2001. Methods for the qualitative evaluation of lexical association measures. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 188–195, Morristown, NJ, USA. Association for Computational Linguistics.

Fabri, Ray. 2009. Compounding and adjective-noun compounds in Maltese. In Comrie, Bernard, Ray Fabri, Elizabeth Hume, Manwel Mifsud, Thomas Stolz, and Martine Vanhove, editors, *Introducing Maltese Linguistics*, volume 113 of *Studies in Language Companion Series*. John Benjamins.

Fazly, Afsaneh and Suzanne Stevenson. 2006. Automatically constructing a lexicon of verb phrase idiomatic combinations. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 337–344.

Glinert, Lewis. 1989. *The Grammar of Modern Hebrew*. Cambridge University Press, Cambridge.

Itai, Alon and Shuly Wintner. 2008. Language resources for Hebrew. *Language Resources and Evaluation*, 42:75–98, March.

Katz, Graham and Eugenie Giesbrecht. 2006. Automatic identification of non-compositional multiword expressions using latent semantic analysis. In *Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, pages 12–19, Sydney, Australia, July. Association for Computational Linguistics.

Levi, Judith N. 1976. A semantic analysis of Hebrew compound nominals. In Cole, Peter, editor, *Studies in Modern Hebrew Syntax and Semantics*, number 32 in North-Holland Linguistic Series, pages 9–55. North-Holland, Amsterdam.

Manning, Christopher D. and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*. The MIT Press, Cambridge, Mass.

Oflazer, Kemal, Özlem Çetinoğlu, and Bilge Say. 2004. Integrating morphology with multi-word expression processing in Turkish. In Tanaka, Takaaki, Aline Villavicencio, Francis Bond, and Anna Korhonen, editors, *Second ACL Workshop on Multiword Expressions: Integrating Processing*, pages 64–71, Barcelona, Spain, July. Association for Computational Linguistics.

Ordan, Noam and Shuly Wintner. 2007. Hebrew WordNet: a test case of aligning lexical databases across languages. *International Journal of Translation, special issue on Lexical Resources for Machine Translation*, 19(1).

Pecina, Pavel. 2008. A machine learning approach to multiword expression extraction. In *Proceedings of the LREC Workshop Towards a Shared Task for Multiword Expressions*.

Ramisch, Carlos, Paulo Schreiner, Marco Idiart, and Alline Villavicencio. 2008. An evaluation of methods for the extraction of multiword expressions. In *Proceedings of the LREC Workshop Towards a Shared Task for Multiword Expressions*.

Sag, Ivan, Timothy Baldwin, Francis Bond, Ann Copestake, and Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2002)*, pages 1–15, Mexico City, Mexico.

Tsvetkov, Yulia and Shuly Wintner. 2010a. Automatic acquisition of parallel corpora from websites with dynamic content. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, pages 3389–3392. European Language Resources Association (ELRA), May.

Tsvetkov, Yulia and Shuly Wintner. 2010b. Extraction of multi-word expressions from small parallel corpora. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, August.

Van de Cruys, Tim and Begoña Villada Moirón. 2007. Semantics-based multiword expression extraction. In *Proceedings of the Workshop on A Broader Perspective on Multiword Expressions*, pages 25–32, Prague, Czech Republic, June. Association for Computational Linguistics.

Villada Moirón, Begoña and Jörg Tiedemann. 2006. Identifying idiomatic expressions using automatic word alignment. In *Proceedings of the EACL 2006 Workshop on Multi-word-expressions in a multilingual context*. Association for Computational Linguistics.

Villavicencio, Aline, Francis Bond, Anna Korhonen, and Diana McCarthy. 2005. Introduction to the special issue on multiword expressions: Having a crack at a hard nut. *Computer Speech & Language*, 19(4):365–377.

Wintner, Shuly. 2000. Definiteness in the Hebrew noun phrase. *Journal of Linguistics*, 36:319–363.

Yona, Shlomo and Shuly Wintner. 2008. A finite-state morphological grammar of Hebrew. *Natural Language Engineering*, 14(2):173–190, April.

Robust Measurement and Comparison of Context Similarity for Finding Translation Pairs

Daniel Andrade[†], **Tetsuya Nasukawa**[‡], **Jun'ichi Tsujii**[†]
[†]Department of Computer Science, University of Tokyo
{daniel.andrade, tsujii}@is.s.u-tokyo.ac.jp
[‡]IBM Research - Tokyo
nasukawa@jp.ibm.com

Abstract

In cross-language information retrieval it is often important to align words that are similar in meaning in two corpora written in different languages. Previous research shows that using context similarity to align words is helpful when no dictionary entry is available. We suggest a new method which selects a subset of words (pivot words) associated with a query and then matches these words across languages. To detect word associations, we demonstrate that a new Bayesian method for estimating Point-wise Mutual Information provides improved accuracy. In the second step, matching is done in a novel way that calculates the chance of an accidental overlap of pivot words using the hypergeometric distribution. We implemented a wide variety of previously suggested methods. Testing in two conditions, a small comparable corpora pair and a large but unrelated corpora pair, both written in disparate languages, we show that our approach consistently outperforms the other systems.

1 Introduction

Translating domain-specific, technical terms from one language to another can be challenging because they are often not listed in a general dictionary. The problem is exemplified in cross-lingual information retrieval (Chiao and Zweigenbaum, 2002) restricted to a certain domain. In this case, the user might enter only a few technical terms. However, jargons that appear frequently in the data set but not in general dictionaries, impair the usefulness of such systems. Therefore, various means to extract translation pairs automatically have been proposed. They use different clues, mainly

- Spelling distance or transliterations, which are useful to identify loan words (Koehn and Knight, 2002).
- Context similarity, helpful since two words with identical meaning are often used in similar contexts across languages (Rapp, 1999).

The first type of information is quite specific; it can only be helpful in a few cases, and can thereby engender high-precision systems with low recall, as described for example in (Koehn and Knight, 2002). The latter is more general. It holds for most words including loan words. Usually the context of a word is defined by the words which occur around it (bag-of-words model).

Let us briefly recall the main idea for using context similarity to find translation pairs. First, the degree of association between the query word and all content words is measured with respect to the corpus at hand. The same is done for every possible translation candidate in the target corpus. This way, we can create a feature vector for the query and all its possible translation candidates. We can assume that, for some content words, we have valid translations in a general dictionary, which enables us to compare the vectors across languages. We will designate these content words as pivot words. The query and its translation candidates are then compared using their feature vectors, where each dimension in the feature vector contains the degree of association to

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 19–27,
Beijing, August 2010

one pivot word. We define the degree of association, as a measurement for finding words that co-occur, or which do not co-occur, more often than we would expect by pure chance.[1]

We argue that common ways for comparing similarity vectors across different corpora perform worse because they assume that degree of associations are very similar across languages and can be compared without much preprocessing. We therefore suggest a new robust method including two steps. Given a query word, in the first step we determine the set of pivots that are all positively associated with statistical significance. In the second step, we compare this set of pivots with the set of pivots extracted for a possible translation candidate. For extracting positively associated pivots, we suggest using a new Bayesian method for estimating the critical Pointwise Mutual Information (PMI) value. In the second step, we use a novel measure to compare the sets of extracted pivot words which is based on an estimation of the probability that pivot words overlap by pure chance. Our approach engenders statistically significant improved accuracy for aligning translation pairs, when compared to a variety of previously suggested methods. We confirmed our findings using two very different pairs of comparable corpora for Japanese and English.

In the next section, we review previous related work. In Section 3 we explain our method in detail, and argue that it overcomes subtle weaknesses of several previous efforts. In Section 4, we show with a series of cross-lingual experiments that our method, in some settings, can lead to considerable improvement in accuracy. Subsequently in Section 4.2, we analyze our method in contrast to the baseline by giving two examples. We summarize our findings in Section 5.

2 Related Work

Extracting context similarity for nouns and then matching them across languages to find translation pairs was pioneered in (Rapp, 1999) and (Fung, 1998). The work in (Chiao and Zweigenbaum, 2002), which can be regarded as a variation of (Fung, 1998), uses tf.idf, but suggests to normalize the term frequency by the maximum number of co-occurrences of two words in the corpus. All this work is closely related to our work because they solely consider context similarity, whereas context is defined using a word window. The work in (Rapp, 1999; Fung, 1998; Chiao and Zweigenbaum, 2002) will form the baselines for our experiments in Section 4.[2] This baseline is also similar to the baseline in (Gaussier et al., 2004), which showed that it can be difficult to beat such a feature vector approach.

In principle our method is not restricted to how context is defined; we could also use, for example, modifiers and head words, as in (Garera et al., 2009). Although, we found in a preliminary experiment that using a dependency parser to differentiate between modifiers and head words like in (Garera et al., 2009), instead of a bag-of-words model, in our setting, actually decreased accuracy due to the narrow dependency window. However, our method could be combined with a back-translation step, which is expected to improve translation quality as in (Haghighi et al., 2008), which performs indirectly a back-translation by matching *all* nouns mutually exclusive across corpora. Notably, there also exist promising approaches which use both types of information, spelling distance, and context similarity in a joint framework, see (Haghighi et al., 2008), or (Déjean et al., 2002) which include knowledge of a thesaurus. In our work here, we concentrate on the use of degrees of association as an effective means to extract word translations.

In this application, to measure association robustly, often the Log-Likelihood Ratio (LLR) measurement is suggested (Rapp, 1999; Morin et al., 2007; Chiao and Zweigenbaum, 2002). The occurrence of a word in a document is modeled as a binary random variable. The LLR measurement measures stochastic dependency between

[1]For example "car" and "tire" are expected to have a high (positive) degree of association, and "car" and "apple" is expected to have a high (negative) degree of association.

[2]Notable differences are that we neglected word order, in contrast to (Rapp, 1999), as it is little useful to compare it between Japanese and English. Furthermore in contrast to (Fung, 1998) we use only one translation in the dictionary, which we select by comparing the relative frequencies. We also made a second run of the experiments where we manually selected the correct translations for the first half of the most frequent pivots – Results did not change significantly.

two such random variables (Dunning, 1993), and is known to be equal to Mutual Information that is linearly scaled by the size of the corpus (Moore, 2004). This means it is a measure for how much the occurrence of word A makes the occurrence of word B more likely, which we term positive association, and how much the absence of word A makes the occurrence of word B more likely, which we term negative association. However, our experiments show that only positive association is beneficial for aligning words cross-lingually. In fact, LLR can still be used for extracting positive associations by filtering in a pre-processing step words with possibly negative associations (Moore, 2005). Nevertheless a problem which cannot be easily remedied is that confidence estimates using LLR are unreliable for small sample sizes (Moore, 2004). We suggest a more principled approach that measures from the start only how much the occurrence of word A makes the occurrence of word B more likely, which is designated as Robust PMI.

Another point that is common to (Rapp, 1999; Morin et al., 2007; Chiao and Zweigenbaum, 2002; Garera et al., 2009; Gaussier et al., 2004) is that word association is compared in a fine-grained way, i.e. they compare the degree of association[3] with every pivot word, even when it is low or exceptionally high. They suggest as a comparison measurement Jaccard similarity, Cosine similarity, and the L1 (Manhattan) distance.

3 Our Approach

We presume that rather than similarity between *degree (strength of)* of associations, the *existence* of common word associations is a more reliable measure for word similarity because the degrees of association are difficult to compare for the following reasons:

- **Small differences in the degree of association are not statistically significant**
  Taking, for example, two sample sets from the same corpus, we will in general measure different degrees of association.

- **Differences in sub-domains / sub-topics**
  Corpora sharing the same topic can still differ in sub-topics.

- **Differences in style or language**
  Differences in word usage. [4]

Other information that is used in vector approaches such as that in (Rapp, 1999) is negative association, although negative association is less informative than positive. Therefore, if it is used at all, it should be assigned a much smaller weight.

Our approach caters to these points, by first deciding whether a pivot word is positively associated (with statistical significance) or whether it is not, and then uses solely this information for finding translation pairs in comparable corpora. It is divisible into two steps. In the first, we use a Bayesian estimated Pointwise Mutual Information (PMI) measurement to find the pivots that are positively associated with a certain word with high confidence. In the second step, we compare two words using their associated pivots as features. The similarity of feature sets is calculated using pointwise entropy. The words for which feature sets have high similarity are assumed to be related in meaning.

3.1 Extracting positively associated words – Feature Sets

To measure the degree of positive association between two words x and y, we suggest the use of information about how much the occurrence of word x makes the occurrence of word y more likely. We express this using Pointwise Mutual Information (PMI), which is defined as follows:

$$PMI(x, y) = \log \frac{p(x, y)}{p(x) \cdot p(y)} = \log \frac{p(x|y)}{p(x)} \,.$$

Therein, $p(x)$ is the probability that word x occurs in a document; $p(y)$ is defined analogously. Furthermore, $p(x, y)$ is the probability that both

[3] To clarify terminology, where possible, we will try to distinguish between *association* and *degree of association*. For example word “car” has the *association* “tire”, whereas the *degree of association* with “tire” is a continuous number, like 5.6.

[4] For example, “stop” is not the only word to describe the fact that a car halted.

words occur in the same document. A positive association is given if $p(x|y) > p(x)$. In related works that use the PMI (Morin et al., 2007), these probabilities are simply estimated using relative frequencies, as

$$PMI(x, y) = \log \frac{\frac{f(x,y)}{n}}{\frac{f(x)}{n}\frac{f(y)}{n}} ,$$

where $f(x)$, $f(y)$ is the document frequency of word x and word y, and $f(x, y)$ is the co-occurrence frequency; n is the number of documents. However, using relative frequencies to estimate these probabilities can, for low-frequency words, produce unreliable estimates for PMI (Manning and Schütze, 2002). It is therefore necessary to determine the uncertainty of PMI estimates. The idea of defining confidence intervals over PMI values is not new (Johnson, 2001); however, the problem is that exact calculation is very computationally expensive if the number of documents is large, in which case one can approximate the binomial approximation for example with a Gaussian, which is, however only justified if n is large *and* p, the probability of an occurrence, is not close to zero (Wilcox, 2009). We suggest to define a beta distribution over each probability of the binary events that word x occurs, i.e. $[x]$, and analogously $[x|y]$. It was shown in (Ross, 2003) that a Bayesian estimate for Bernoulli trials using the beta distribution delivers good credibility intervals[5], importantly, when sample sizes are small, or when occurrence probabilities are close to 0. Therefore, we assume that

$$p(x|y) \sim beta(\alpha'_{x|y}, \beta'_{x|y}), p(x) \sim beta(\alpha'_x, \beta'_x)$$

where the parameters for the two beta distributions are set to

$$\begin{aligned}
&\alpha'_{x|y} = f(x, y) + \alpha_{x|y} \, , \\
&\beta'_{x|y} = f(y) - f(x, y) + \beta_{x|y} \, , \text{and} \\
&\alpha'_x = f(x) + \alpha_x, \, \beta'_x = n - f(x) + \beta_x \, .
\end{aligned}$$

Prior information related to $p(x)$ and the conditional probability $p(x|y)$ can be incorporated by setting the hyper-parameters of the beta-distribtutions.[6] These can, for example, be learned from another unrelated corpora pair and then weighted appropriately by setting $\alpha + \beta$. For our experiments, we use no information beyond the given corpora pair; the conditional priors are therefore set equal to the prior for $p(x)$. Even if we do not know which word x is, we have a notion about $p(x)$ because Zipf's law indicates to us that we should expect it to be small. A crude estimation is therefore the mean word occurrence probability in our corpus as

$$\gamma = \frac{1}{|\text{all words}|} \sum_{x \in \{\text{all words}\}} \frac{f(x)}{n} \, .$$

We give this estimate a total weight of one observation. That is, we set

$$\alpha = \gamma \, , \beta = 1 - \gamma \, .$$

From a practical perspective, this can be interpreted as a smoothing when sample sizes are small, which is often the case for $p(x|y)$. Because we assume that $p(x|y)$ and $p(x)$ are random variables, PMI is consequently also a random variable that is distributed according to a beta distribution ratio.[7] For our experiments, we apply a general sampling strategy. We sample $p(x|y)$ and $p(x)$ independently and then calculate the ratio of times $PMI > 0$ to determine $P(PMI > 0)$.[8] We will refer to this method as Robust PMI (RPMI).

Finally we can calculate, for any word x, the set of pivot words which have most likely a positive association with word x. We require that this set be statistically significant: the probability of one or more words being not a positive association is smaller than a certain p-value.[9]

[5]In the Bayesian notation we refer here to credibility intervals instead of confidence intervals.

[6]The hyper-parameters α and β, can be intuitively interpreted in terms of document frequency. For example α_x is the number of times we belief the word x occurs, and β_x the number of times we belief that x does not occur in a corpus. Analogously $\alpha_{x|y}$ and $\beta_{x|y}$ can be interpreted with respect to the subset of the corpus where the word y occurs, instead of the whole corpus. Note however, that α and β do not necessarily have to be integers.

[7]The resulting distribution for the general case of a beta distribution ratio was derived in (Pham-Gia, 2000). Unfortunately, it involves the calculation of a Gauss hyper-geometric function that is computationally expensive for large n.

[8]For experiments, we used $100,000$ samples for each estimate of $P(PMI > 0)$.

[9]We set, for all of our experiments, the p-value to 0.01.

As an alternative for determining the probability of a positive association using $P(PMI > 0)$, we calculate LLR and assume that approximately $LLR \sim \chi^2$ with one degree of freedom (Dunning, 1993). Furthermore, to ensure that only positive association counts, we set the probability to zero if $p(x, y) < p(x) \cdot p(y)$, where the probabilities are estimated using relative frequencies (Moore, 2005). We refer to this as LLR(P); lacking this correction, it is LLR.

3.2 Comparing Word Feature Sets Across Corpora

So far, we have explained a robust means to extract the pivot words that have a positive association with the query. The next task is to find a sensible way to use these pivots to compare the query with candidates from the target corpus. A simple means to match a candidate with a query is to see how many pivots they have in common, i.e. using the matching coefficient (Manning and Schütze, 2002) to score candidates. This similarity measure produces a reasonable result, as we will show in the experiment section; however, in our error analysis, we found out that this gives a bias to candidates with higher frequencies, which is explainable as follows. Assuming that a word A has a fixed number of pivots that are positively associated, then depending on the sample size—the document frequency in the corpus—not all of these are statistically significant. Therefore, not all true positive associations are included in the feature set to avoid possible noise. If the document frequency increases, then we can extract more statistically significant positive associations and the cardinality of the feature set increases. This consequently increases the likelihood of having more pivots that overlap with pivots from the query's feature set. For example, imagine two candidate words A and B, for which feature sets of both include the feature set of the query, i.e. a complete match, however A's feature set is much larger than B's feature set. In this case, the information conveyed by having a complete match with the query word's feature set is lower in the case of A's feature set than in case of B's feature set. Therefore, we suggest its use as a basis of our similarity measure, the degree of pointwise entropy of having an estimate of m matches, as

$$\text{Information}(m, q, c) = -\log(P(matches = m)).$$

Therein, $P(matches = m)$ is the likelihood that a candidate word with c pivots has m matches with the query word, which has q pivots. Letting w be the total number of pivot words, we can then calculate that the probability that the candidate with c pivots was selected by chance

$$P(matches = m) = \frac{\binom{q}{m} \cdot \binom{w-q}{c-m}}{\binom{w}{c}}.$$

Note that this probability equals a hypergeometric distribution.[10] The smaller $P(matches = m)$ is, the less likely it is that we obtain m matches by pure chance. In other words, if $P(matches = m)$ is very small, m matches are more than we would expect to occur by pure chance.[11]

Alternatively, in our experiments, we also consider standard similarity measurements (Manning and Schütze, 2002) such as the Tanimoto coefficient, which also lowers the score of candidates that have larger feature sets.

4 Experiments

In our experiments, we specifically examine translating nouns, mostly technical terms, which occur in complaints about cars collected by the Japanese Ministry of Land, Infrastructure, Transport and Tourism (MLIT)[12], and in complaints about cars collected by the USA National Highway Traffic Safety Administration (NHTSA)[13]. We create for each data collection a corpus for which a document corresponds to one car customer reporting a certain problem in free text. The complaints are, in general, only a few sentences long.

[10] $\binom{q}{m}$ is the number of possible combinations of pivots which the candidate has in common with the query. Therefore, $\binom{q}{m} \cdot \binom{w-q}{c-m}$ is the number of possible different feature sets that the candidate can have such that it shares m common pivots with the query. Furthermore, $\binom{w}{c}$ is the total number of possible feature sets the candidate can have.

[11] The discussion is simplified here. It can also be that $P(matches = m)$ is very small, if there are less occurrences of m that we would expect to occur by pure chance. However, this case can be easily identified by looking at the gradient of $P(matches = m)$.

[12] http://www.mlit.go.jp/jidosha/carinf/rcl/defects.html

[13] http://www-odi.nhtsa.dot.gov/downloads/index.cfm

To verify whether our results can be generalized over other pairs of comparable corpora, we additionally made experiments using two corpora extracted from articles of Mainichi Shinbun, a Japanese newspaper, in 1995 and English articles from Reuters in 1997. There are two notable differences between those two pairs of corpora: the content is much less comparable, Mainichi reports more national news than world news, and secondly, Mainichi and Reuters corpora are much larger than MLIT/NHTSA.[14]

For both corpora pairs, we extracted a gold-standard semi-automatically by looking at Japanese nouns and their translations with document frequency of at least 50 for MLIT/NHTSA, and 100 for Mainichi/Reuters. As a dictionary we used the Japanese-English dictionary JMDic[15]. In general, we preferred domain-specific terms over very general terms, i.e. for example for MLIT/NHTSA the noun 噴射 "injection" was preferred over 取り付け "installation". We extracted 100 noun pairs for MLIT/NHTSA and Mainichi/Reuters, each. Each Japanese noun which is listed in the gold-standard forms a query which is input into our system. The resulting ranking of the translation candidates is automatically evaluated using the gold-standard. Therefore, synonyms that are not listed in the gold standard are not recognized, engendering a conservative estimation of the translation accuracy. Because all methods return a ranked list of translation candidates, the accuracy is measured using the rank of the translation listed in the gold-standard.[16] The Japanese corpora are preprocessed with MeCab (Kudo et al., 2004); the English corpora with Stepp Tagger (Tsuruoka et al., 2005) and Lemmatizer (Okazaki et al., 2008). As a dictionary we use the Japanese-English dictionary JMDic[17]. In line with related work (Gaussier et al., 2004), we remove a word pair (Japanese noun s, English noun t) from the dictionary, if s occurs in the gold-standard. Afterwards we define the pivot words by consulting the remaining dictionary.

4.1 Crosslingual Experiment

We compare our approach used for extracting cross-lingual translation pairs against several baselines. We compare to LLR + Manhattan (Rapp, 1999) and our variation LLR(P) + Manhattan. Additionally, we compare TFIDF(MSO) + Cosine, which is the TFIDF measure, whereas the Term Frequency is normalized using the maximal word frequency and the cosine similarity for comparison suggested in (Fung, 1998). Furthermore, we implemented two variations of this, TFIDF(MPO) + Cosine and TFIDF(MPO) + Jaccard coefficient, which were suggested in (Chiao and Zweigenbaum, 2002). In fact, TFIDF(MPO) is the TFIDF measure, whereas the Term Frequency is normalized using the maximal word pair frequency. The results are displayed in Figure 1. Our approach clearly outperforms all baselines; notably it has Top 1 accuracy of 0.14 and Top 20 accuracy of 0.55, which is much better than that for the best baseline, which is 0.11 and 0.44, respectively.

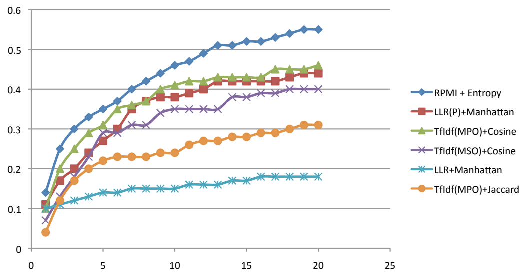


Figure 1: Crosslingual Experiment MLIT/NHTSA – Percentile Ranking of RPMI + Entropy Against Various Previous Suggested Methods.

We next leave the proposed framework constant, but change the mode of estimating positive associations and the way to match feature sets. As alternatives for estimating the probability that there is a positive association, we test LLR(P) and LLR. As alternatives for comparing feature sets, we investigate the matching coefficient (match), cosine similarity (cosine), Tanimoto coefficient (tani), and overlap coefficient

[14]MLIT/MLIT has each 20,000 documents. Mainichi/Reuters corpora 75,935 and 148,043 documents, respectively.

[15]http://www.csse.monash.edu.au/ jwb/edict_doc.html

[16]In cases for which there are several translations listed for one word, the rank of the first is used.

[17]http://www.csse.monash.edu.au/ jwb/edict_doc.html

(over) (Manning and Schütze, 2002). The result of every combination is displayed concisely in Table 1 using the median rank[18]. The cases in which the median ranks are close to RPMI + Entropy are magnified in Table 2. We can see there that RPMI + Entropy, and LLR(P) + Entropy perform nearly equally. All other combinations perform worse, especially in Top 1 accuracy. Finally, LLR(P) presents a clear edge over LLR, which suggests that indeed only positive associations seem to matter in a cross-lingual setting.

	Entropy	Match	Cosine	Tani	Over
RPMI	13.0	17.0	24.0	37.5	36.0
LLR(P)	16.0	15.0	22.5	34.0	25.5
LLR	23.5	22.0	27.5	50.5	50.0

Table 1: Crosslingual experiment MLIT/NHTSA – Evaluation matrix showing the median ranks of several combinations of association and similarity measures.

	Top 1	Top 10	Top 20
RPMI + Entropy	0.14	0.46	0.55
RPMI + Matching	0.08	0.41	0.57
LLR(P) + Entropy	0.14	0.46	0.55
LLR(P) + Matching	0.08	0.44	0.55

Table 2: Accuracies for crosslingual experiment MLIT/NHTSA.

Finally we conduct an another experiment using the corpora pair Mainichi/Reuters which is quite different from MLIT/NHTSA. When comparing to the best baselines in Table 3 we see that our approach again performs best. Furthermore, the experiments displayed in Table 4 suggest that Robust PMI and pointwise entropy are better choices for positive association measurement and similarity measurement, respectively. We can see that

	Top 1	Top 10	Top 20
RPMI + Entropy	0.15	0.38	0.46
LLR(P) + Manhattan	0.10	0.26	0.33
TFIDF(MPO) + Cos	0.05	0.12	0.18

Table 3: Accuracies for crosslingual experiment Mainichi/Reuters – Comparison to best baselines.

	Top 1	Top 10	Top 20
RPMI + Entropy	0.15	0.38	0.46
RPMI + Matching	0.08	0.30	0.35
LLR(P) + Entropy	0.13	0.36	0.47
LLR(P) + Matching	0.08	0.29	0.37

Table 4: Accuracies for crosslingual experiment Mainichi/Reuters – Comparison to alternatives.

the overall best baseline turns out to be LLR(P) + Manhattan. Comparing the rank from each word from the gold-standard pairwise, we see that our approach, RPMI + Entropy, is significantly better than this baseline in MLIT/NHTSA as well as in Mainichi/Reuters.[19]

4.2 Analysis

In this section, we provide two representative examples extracted from the previous experiments which sheds light into a weakness of the standard feature vector approach which was used as a baseline before. The two example queries and the corresponding responses of LLR(P) + Manhattan and our approach are listed in Table 5. Furthermore in Table 6 we list the pivot words with the highest degree of association (here LLR values) for the query and its correct translation. We can see that a query and its translation shares some pivots which are associated with statistical significance[20]. However it also illustrates that the actual LLR value is less insightful and can hardly be compared across these two corpora.

Let us analyze the two examples in more detail. In Table 6, we see that the first query ギア "gear"[21] is highly associated with 入れる "shift". However, on the English side we see that gear is most highly associated with the pivot word gear. Note that here the word gear is also a pivot word corresponding to the Japanese pivot word 歯車 "gear (wheel)".[22] Since in English the word gear (shift) and gear (wheel) is polysemous, the surface forms are the same leading to a high LLR value of

[18]A median rank of i, means that 50% of the correct translations have a rank higher than i.

[19]Using pairwise test with p-value 0.05.

[20]Note that for example, an LLR value bigger than 11.0 means the chances that there is no association is smaller than 0.001 using that $LLR \sim \chi^2$.

[21]For a Japanese word, we write the English translation which is *appropriate in our context*, immediately after it.

[22]In other words, we have the entry (歯車, gear) in our dictionary but not the entry (ギア, gear). The first pair is used as a pivot, the latter word pair is what we try to find.

gear. Finally, the second example query ペダル "pedal" shows that words which, not necessarily always, but very often co-occur, can cause relatively high LLR values. The Japanese verb 踏む "to press" is associated with ペダル with a high LLR value – 4 times higher than 戻る "return" – which is not reflected on the English side. In summary, we can see that in both cases the degree of associations are rather different, and cannot be compared without preprocessing. However, it is also apparent that in both examples a simple L1 normalization of the degree of associations does *not* lead to more similarity, since the relative differences remain.

ギア "gear"		
Method	**Top 3 candidates**	**Rank**
baseline	jolt, lever, design	284
filtering	reverse, gear, lever	2
ペダル "pedal"		
Method	**Top 3 candidates**	**Rank**
baseline	mj, toyota, action	176
filtering	pedal, situation, occasion	1

Table 5: List of translation suggestions using LLR(P) + Manhattan (baseline) and our method (filtering). The third column shows the rank of the correct translation.

ギア		**gear**	
Pivots	**LLR(P)**	**Pivots**	**LLR(P)**
入る "shift"	154	gear	7064
入れる "shift"	144	shift	1270
抜ける "come out"	116	reverse	314
ペダル		**pedal**	
Pivots	**LLR(P)**	**Pivots**	**LLR(P)**
踏む "press"	628	floor	1150
戻る "return"	175	stop	573
足 "foot"	127	press	235

Table 6: Shows the three pivot words which have the highest degree of association with the query (left side) and the correct translation (right side).

5 Conclusions

We introduced a new method to compare context similarity across comparable corpora using a Bayesian estimate for PMI (Robust PMI) to extract positive associations and a similarity measurement based on the hypergeometric distribution (measuring pointwise entropy). Our experiments show that, for finding cross-lingual translations, the assumption that words with similar meaning share positive associations with the same words is more appropriate than the assumption that the degree of association is similar. Our approach increases Top 1 and Top 20 accuracy of up to 50% and 39% respectively, when compared to several previous methods. We also analyzed the two components of our method separately. In general, Robust PMI yields slightly better performance than the popular LLR, and, in contrast to LLR, allows to extract positive associations as well as to include prior information in a principled way. Pointwise entropy for comparing feature sets cross-lingually improved the translation accuracy clearly when compared with standard similarity measurements.

Acknowledgment

We thank Dr. Naoaki Okazaki and the anonymous reviewers for their helpful comments. Furthermore we thank Daisuke Takuma, IBM Research - Tokyo, for mentioning previous work on statistical corrections for PMI. This work was partially supported by Grant-in-Aid for Specially Promoted Research (MEXT, Japan). The first author is supported by the MEXT Scholarship and by an IBM PhD Scholarship Award.

References

Chiao, Y.C. and P. Zweigenbaum. 2002. Looking for candidate translational equivalents in specialized, comparable corpora. In *Proceedings of the International Conference on Computational Linguistics*, pages 1–5. International Committee on Computational Linguistics.

Déjean, H., É. Gaussier, and F. Sadat. 2002. An approach based on multilingual thesauri and model combination for bilingual lexicon extraction. In *Proceedings of the International Conference on Computational Linguistics*, pages 1–7. International Committee on Computational Linguistics.

Dunning, T. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Fung, P. 1998. A statistical view on bilingual lexicon extraction: from parallel corpora to non-parallel corpora. *Lecture Notes in Computer Science*, 1529:1–17.

Garera, N., C. Callison-Burch, and D. Yarowsky. 2009. Improving translation lexicon induction from monolingual corpora via dependency contexts and part-of-speech equivalences. In *Proceedings of the Conference on Computational Natural Language Learning*, pages 129–137. Association for Computational Linguistics.

Gaussier, E., J.M. Renders, I. Matveeva, C. Goutte, and H. Dejean. 2004. A geometric view on bilingual lexicon extraction from comparable corpora. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 526–533. Association for Computational Linguistics.

Haghighi, A., P. Liang, T. Berg-Kirkpatrick, and D. Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 771–779. Association for Computational Linguistics.

Johnson, M. 2001. Trading recall for precision with confidence-sets. Technical report, Brown University.

Koehn, P. and K. Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of ACL Workshop on Unsupervised Lexical Acquisition*, volume 34, pages 9–16. Association for Computational Linguistics.

Kudo, T., K. Yamamoto, and Y. Matsumoto. 2004. Applying conditional random fields to Japanese morphological analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 230–237. Association for Computational Linguistics.

Manning, C.D. and H. Schütze. 2002. *Foundations of Statistical Natural Language Processing*. MIT Press.

Moore, R.C. 2004. On log-likelihood-ratios and the significance of rare events. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 333–340. Association for Computational Linguistics.

Moore, R.C. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 81–88. Association for Computational Linguistics.

Morin, E., B. Daille, K. Takeuchi, and K. Kageura. 2007. Bilingual terminology mining-using brain, not brawn comparable corpora. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, volume 45, pages 664–671. Association for Computational Linguistics.

Okazaki, N., Y. Tsuruoka, S. Ananiadou, and J. Tsujii. 2008. A discriminative candidate generator for string transformations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 447–456. Association for Computational Linguistics.

Pham-Gia, T. 2000. Distributions of the ratios of independent beta variables and applications. *Communications in Statistics. Theory and Methods*, 29(12):2693–2715.

Rapp, R. 1999. Automatic identification of word translations from unrelated English and German corpora. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 519–526. Association for Computational Linguistics.

Ross, T.D. 2003. Accurate confidence intervals for binomial proportion and Poisson rate estimation. *Computers in Biology and Medicine*, 33(6):509–531.

Tsuruoka, Y., Y. Tateishi, J. Kim, T. Ohta, J. McNaught, S. Ananiadou, and J. Tsujii. 2005. Developing a robust part-of-speech tagger for biomedical text. *Lecture Notes in Computer Science*, 3746:382–392.

Wilcox, R.R. 2009. *Basic Statistics: Understanding Conventional Methods and Modern Insights*. Oxford University Press.

Multilingual Subjectivity: Are More Languages Better?

Carmen Banea, Rada Mihalcea
Department of Computer Science
University of North Texas
carmenbanea@my.unt.edu
rada@cs.unt.edu

Janyce Wiebe
Department of Computer Science
University of Pittsburgh
wiebe@cs.pitt.edu

Abstract

While subjectivity related research in other languages has increased, most of the work focuses on single languages. This paper explores the integration of features originating from multiple languages into a machine learning approach to subjectivity analysis, and aims to show that this enriched feature set provides for more effective modeling for the source as well as the target languages. We show not only that we are able to achieve over 75% macro accuracy in all of the six languages we experiment with, but also that by using features drawn from multiple languages we can construct high-precision meta-classifiers with a precision of over 83%.

1 Introduction

Following the terminology proposed by (Wiebe et al., 2005), subjectivity and sentiment analysis focuses on the automatic identification of private states, such as opinions, emotions, sentiments, evaluations, beliefs, and speculations in natural language. While subjectivity classification labels text as either subjective or objective, sentiment or polarity classification adds an additional level of granularity, by further classifying subjective text as either positive, negative or neutral.

To date, a large number of text processing applications have used techniques for automatic sentiment and subjectivity analysis, including automatic expressive text-to-speech synthesis (Alm et al., 1990), tracking sentiment timelines in on-line forums and news (Balog et al., 2006; Lloyd et al., 2005), and mining opinions from product reviews (Hu and Liu, 2004). In many natural language processing tasks, subjectivity and sentiment classification has been used as a first phase filtering to generate more viable data. Research that benefited from this additional layering ranges from question answering (Yu and Hatzivassiloglou, 2003), to conversation summarization (Carenini et al., 2008), and text semantic analysis (Wiebe and Mihalcea, 2006; Esuli and Sebastiani, 2006a).

Although subjectivity tends to be preserved across languages – see the manual study in (Mihalcea et al., 2007), (Banea et al., 2008) hypothesize that subjectivity is expressed differently in various languages due to lexicalization, formal versus informal markers, etc. Based on this observation, our research seeks to answer the following questions. First, can we *reliably* predict sentence-level subjectivity in languages other than English, by leveraging on a manually annotated English dataset? Second, can we improve the English subjectivity classification by expanding the feature space through the use of multilingual data? Similarly, can we also improve the classifiers in the other target languages? Finally, third, can we benefit from the multilingual subjectivity space and build a high-precision subjectivity classifier that could be used to generate subjectivity datasets in the target languages?

The paper is organized as follows. We introduce the datasets and the general framework in Section 2. Sections 3, 4, and 5 address in turn each of the three research questions mentioned above. Section 6 describes related literature in the area of multilingual subjectivity. Finally, we draw our conclusions in Section 7.

2 Multilingual Datasets

Corpora that are manually annotated for subjectivity, polarity, or emotion, are available in only select languages, since they require a considerable amount of human effort. Due to this impediment, the focus of this paper is to create a method for extrapolating subjectivity data devel-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 28–36,
Beijing, August 2010

SubjP	SubjR	SubjF	ObjP	ObjR	ObjF	AllP	AllR	AllF
90.4%	34.2%	46.6%	82.4%	30.7%	44.7%	86.7%	32.6%	47.4%

Table 1: Results obtained with a rule-based subjectivity classifier on the MPQA corpus (Wiebe and Riloff, 2005)

oped in a source language and to transfer it to other languages. Multilingual feature spaces are generated to create even better subjectivity classifiers, outperforming those trained on the individual languages alone.

We use the Multi-Perspective Question Answering (MPQA) corpus, consisting of 535 English-language news articles from a variety of sources, manually annotated for subjectivity (Wiebe et al., 2005). Although the corpus is annotated at the clause and phrase levels, we use the sentence-level annotations associated with the dataset in (Wiebe and Riloff, 2005). A sentence is labeled as subjective if it has at least one private state of strength medium or higher. Otherwise the sentence is labeled as objective. From the approximately 9700 sentences in this corpus, 55% of them are labeled as subjective, while the rest are objective. Therefore, 55% represents the majority baseline on this corpus. (Wiebe and Riloff, 2005) apply both a subjective and an objective rule-based classifier to the MPQA corpus data and obtain the results presented in Table 1.[1]

In order to generate parallel corpora to MPQA in other languages, we rely on the method we proposed in (Banea et al., 2008). We experiment with five languages other than English (*En*), namely Arabic (*Ar*), French (*Fr*), German (*De*), Romanian (*Ro*) and Spanish (*Es*). Our choice of languages is motivated by several reasons. First, we wanted languages that are highly lexicalized and have clear word delimitations. Second, we were interested to cover languages that are similar to English as well as languages with a completely different etymology. Consideration was given to include Asian languages, such as Chinese or Japanese, but the fact that their script without word-segmentation preprocessing does not directly map to words was a deterrent. Finally, another limitation on our choice of languages is the need for a publicly available machine translation system between the source language and each of the target languages.

We construct a subjectivity annotated corpus for each of the five languages by using machine translation to transfer the source language data into the target language. We then project the original sentence level English subjectivity labeling onto the target data. For all languages, other than Romanian, we use the Google Translate service,[2] a publicly available machine translation engine based on statistical models. The reason Romanian is not included in this group is that, at the time when we performed the first experiments, Google Translate did not provide a translation service for this language. Instead, we used an alternative statistical translation system called LanguageWeaver,[3] which was commercially available, and which the company kindly allowed us to use for research purposes.

The raw corpora in the five target languages are available for download at http://lit.csci.unt.edu/index.php/Downloads, while the English MPQA corpus can be obtained from http://www.cs.pitt.edu/mpqa.

Given the specifics of each language, we employ several preprocessing techniques. For Romanian, French, English, German and Spanish, we remove all the diacritics, numbers and punctuation marks except - and '. The exceptions are motivated by the fact that they may mark contractions, such as En: *it's* or Ro: *s-ar* (*may be*), and the component words may not be resolved to the correct forms. For Arabic, although it has a different encoding, we wanted to make sure to treat it in a way similar to the languages with a Roman

[1]For the purpose of this paper we follow this abbreviation style: *Subj* stands for subjective, *Obj* stands for objective, and *All* represents overall macro measures, computed over the subjective and objective classes; *P*, *R*, *F*, and *MAcc* correspond to precision, recall, F-measure, and macro-accuracy, respectively.

[2]http://www.google.com/translate_t

[3]http://www.languageweaver.com/

alphabet. We therefore use a library[4] that maps Arabic script to a space of Roman-alphabet letters supplemented with punctuation marks so that they can allow for additional dimensionality.

Once the corpora are preprocessed, each sentence is defined by six views: one in the original source language (English), and five obtained through automatic translation in each of the target languages. Multiple datasets that cover all possible combinations of six languages taken one through six (a total of 63 combinations) are generated. These datasets feature a vector for each sentence present in MPQA (approximately 9700). The vector contains only unigram features in one language for a monolingual dataset. For a multilingual dataset, the vector represents a cumulation of monolingual unigram features extracted from each view of the sentence. For example, one of the combinations of six taken three is Arabic-German-English. For this combination, the vector is composed of unigram features extracted from each of the Arabic, German and English translations of the sentence.

We perform ten-fold cross validation and train Naïve Bayes classifiers with feature selection on each dataset combination. The top 20% of the features present in the training data are retained. For datasets resulting from combinations of all languages taken one, the classifiers are monolingual classifiers. All other classifiers are multilingual, and their feature space increases with each additional language added. Expanding the feature set by encompassing a group of languages enables us to provide an answer to two problems that can appear due to data sparseness. First, enough training data may not be available in the monolingual corpus alone in order to correctly infer labeling based on statistical measures. Second, features appearing in the monolingual test set may not be present in the training set and therefore their information cannot be used to generate a correct classification.

Both of these problems are further explained through the examples below, where we make the simplifying assumption that the words in italics are the only potential carriers of subjective content, and that, without them, their surrounding contexts would be objective. Therefore, their association with an either objective or subjective meaning imparts to the entire segment the same labeling upon classification.

To explore the first sparseness problem, let us consider the following two examples extracted from the English version of the MPQA dataset, followed by their machine translations in German:

> "En 1: rights group Amnesty International said it was *concerned* about the high risk of violence in the aftermath"
> "En 2: official said that US diplomats to countries *concerned* are authorized to explain to these countries"
> "De 1: Amnesty International sagte, es sei *besorgt* über das hohe Risiko von Gewalt in der Folgezeit"
> "De 2: Beamte sagte, dass US-Diplomaten *betroffenen* Länder berechtigt sind, diese Länder zu erklären"

We focus our discussion on the word *concerned*, which in the first example is used in its subjective sense, while in the second it carries an objective meaning (as it refers to a group of countries exhibiting a particular feature defined earlier on in the context). The words in italics in the German contexts represent the translations of *concerned* into German, which are functionally different as they are shaped by their surrounding context. By training a classifier on the English examples alone, under the data sparseness paradigm, the machine learning model may not differentiate between the word's objective and subjective uses when predicting a label for the entire sentence. However, appending the German translation to the examples generates additional dimensions for this model and allows the classifier to potentially distinguish between the senses and provide the correct sentence label.

For the second problem, let us consider two other examples from the English MPQA and their respective translations into Romanian:

> "En 3: could secure concessions on Taiwan in return for *supporting* Bush on issues such as anti-terrorism and"

[4]Lingua::AR::Word PERL library.

Lang	SubjP	SubjR	SubjF	ObjP	ObjR	ObjF	AllP	AllR	AllF	MAcc
En	74.01%	**83.64%**	**78.53%**	**75.89%**	63.68%	**69.25%**	**74.95%**	**73.66%**	**73.89%**	**74.72%**
Ro	73.50%	82.06%	77.54%	74.08%	63.40%	68.33%	73.79%	72.73%	72.94%	73.72%
Es	**74.02%**	82.84%	78.19%	75.11%	**64.05%**	69.14%	74.57%	73.44%	73.66%	74.44%
Fr	73.83%	83.03%	78.16%	75.19%	63.61%	68.92%	74.51%	73.32%	73.54%	74.35%
De	73.26%	83.49%	78.04%	75.32%	62.30%	68.19%	74.29%	72.90%	73.12%	74.02%
Ar	71.98%	81.47%	76.43%	72.62%	60.78%	66.17%	72.30%	71.13%	71.30%	72.22%

Table 2: Naïve Bayes learners trained on six individual languages

> "En 4: to the potential for change from within America. *Supporting* our schools and community centres is a good"
>
> "Ro 3: ar putea asigura concesii cu privire la Taiwan, în schimb pentru *susţinerea* lui Bush pe probleme cum ar fi anti-terorismului şi"
>
> "Ro 4: la potenţialul de schimbare din interiorul Americii. *Sprijinirea* şcolile noastre şi centre de comunitate este un bun"

In this case, *supporting* is used in both English examples in senses that are both subjective; the word is, however, translated into Romanian through two synonyms, namely *susţinerea* and *sprijinirea*. Let us assume that sufficient training examples are available to strengthen a link between *supporting* and *susţinerea*, and the classifier is presented with a context containing *sprijinirea*, unseen in the training data. A multilingual classifier may be able to predict a label for the context using the co-occurrence metrics based on *supporting* and extrapolate a label when the context contains both the English word and its translation into Romanian as *sprijinirea*. For a monolingual classifier, such an inference is not possible, and the feature is discarded. Therefore a multi-lingual classifier model may gain additional strength from co-occurring words across languages.

3 Question 1

Can we reliably predict sentence-level subjectivity in languages other than English, by leveraging on a manually annotated English dataset?

In (Banea et al., 2008), we explored several methods for porting subjectivity annotated data from a source language (English) to a target language (Romanian and Spanish). Here, we focus on the transfer of manually annotated corpora through the usage of machine translation by projecting the original sentence level annotations onto the generated parallel text in the target language. Our aim is not to improve on that method, but rather to verify that the results are reliable across a number of languages. Therefore, we conduct this experiment in several additional languages, namely French, German and Arabic, and compare the results with those obtained for Spanish and Romanian.

Table 2 shows the results obtained using Naïve Bayes classifiers trained in each language individually, with a macro accuracy ranging from 71.30% (for Arabic) to 73.89% (for English).[5] As expected, the English machine learner outperforms those trained on other languages, as the original language of the annotations is English. However, it is worth noting that all measures do not deviate by more than 3.27%, implying that classifiers built using this technique exhibit a consistent behavior across languages.

4 Question 2

Can we improve the English subjectivity classification by expanding the feature space through the use of multilingual data? Similarly, can we also improve the classifiers in the other target languages?

We now turn towards investigating the impact on subjectivity classification of an expanded feature space through the inclusion of multilingual data. In order to methodically assess classifier behavior, we generate multiple datasets containing all pos-

[5] Note that the experiments conducted in (Banea et al., 2008) were made on a different test set, and thus the results are not directly comparable across the two papers.

No lang	SubjP	SubjR	SubjF	ObjP	ObjR	ObjF	AllP	AllR	AllF
1	73.43%	82.76%	77.82%	74.70%	62.97%	68.33%	74.07%	72.86%	73.08%
2	74.59%	83.14%	78.63%	75.70%	64.97%	69.92%	75.15%	74.05%	74.28%
3	75.04%	83.27%	78.94%	76.06%	65.75%	70.53%	75.55%	74.51%	74.74%
4	75.26%	83.36%	79.10%	76.26%	66.10%	70.82%	75.76%	74.73%	74.96%
5	75.38%	83.45%	79.21%	76.41%	66.29%	70.99%	75.90%	74.87%	75.10%
6	**75.43%**	**83.66%**	**79.33%**	**76.64%**	**66.30%**	**71.10%**	**76.04%**	**74.98%**	**75.21%**

Table 3: Average measures for a particular number of languages in a combination (from one through six) for Naïve Bayes classifiers using a multilingual space

sible combinations of one through six languages, as described in Section 2. We then train Naïve Bayes learners on the multilingual data and average our results per each group comprised of a particular number of languages. For example, for one language, we have the six individual classifiers described in Section 3; for the group of three languages, the average is calculated over 20 possible combinations; and so on.

Table 3 shows the results of this experiment. We can see that the overall F-measure increases from 73.08% – which is the average over one language – to 75.21% when all languages are taken into consideration (8.6% error reduction). We measured the statistical significance of these results by considering on one side the predictions made by the best performing classifier for one language (i.e., English), and on the other side the predictions made by the classifier trained on the multilingual space composed of all six languages. Using a paired t-test, the improvement was found to be significant at $p = 0.001$. It is worth mentioning that both the subjective and the objective precision measures increase to 75% when more than 3 languages are considered, while the overall recall level stays constant at 74%.

To verify that the improvement is due indeed to the addition of multilingual features, and it is not a characteristic of the classifier, we also tested two other classifiers, namely KNN and Rocchio. Figure 1 shows the average macro-accuracies obtained with these classifiers. For all the classifiers, the accuracies of the multilingual combinations exhibit an increasing trend, as a larger number of languages is used to predict the subjectivity annotations. The Naïve Bayes algorithm has the best performance, and a relative error rate reduction in accuracy of 8.25% for the grouping formed of six languages versus one, while KNN and Rocchio exhibit an error rate reduction of 5.82% and 9.45%, respectively. All of these reductions are statistically significant.

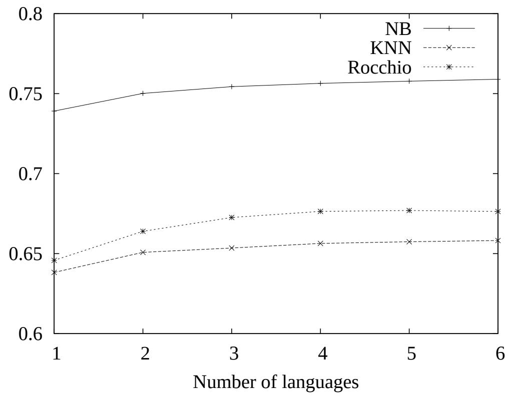


Figure 1: Average Macro-Accuracy per group of languages (combinations of 6 taken one through six)

In order to assess how the proposed multilingual expansion improves on the individual language classifiers, we select one language at a time to be the reference, and then compute the average accuracies of the Naïve Bayes learner across all the language groupings (from one through six) that contain the language. The results from this experiment are illustrated in Figure 2. The baseline in this case is represented by the accuracy obtained with a classifier trained on only one language (this corresponds to 1 on the X-axis). As more languages are added to the feature space, we notice a steady improvement in performance. When the language of reference is Arabic, we obtain an error reduction of 15.27%; 9.04% for Ro-

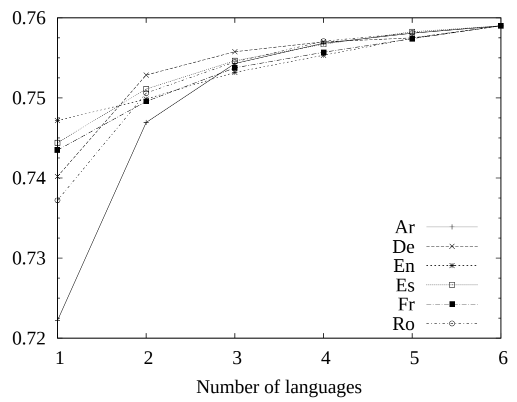


Figure 2: Average macro-accuracy progression relative to a given language

manian; 7.80% for German; 6.44% for French; 6.06% for Spanish; and 4.90 % for English. Even if the improvements seem minor, they are consistent, and the use of a multilingual feature set enables every language to reach a higher accuracy than individually attainable.

In terms of the best classifiers obtained for each grouping of one through six, English provides the best accuracy among individual classifiers (74.71%). When considering all possible combinations of six classifiers taken two, German and Spanish provide the best results, at 75.67%. Upon considering an additional language to the mix, the addition of Romanian to the German-Spanish classifier further improves the accuracy to 76.06%. Next, the addition of Arabic results in the best performing overall classifier, with an accuracy of 76.22%. Upon adding supplemental languages, such as English or French, no further improvements are obtained. We believe this is the case because German and Spanish are able to expand the dimensionality conferred by English alone, while at the same time generating a more orthogonal space. Incrementally, Romanian and Arabic are able to provide high quality features for the classification task. This behavior suggests that languages that are somewhat further apart are more useful for multilingual subjectivity classification than intermediary languages.

5 Question 3

Can we train a high precision classifier with a good recall level which could be used to generate subjectivity datasets in the target languages?

Since we showed that the inclusion of multilingual information improves the performance of subjectivity classifiers for all the languages involved, we further explore how the classifiers' predictions can be combined in order to generate high-precision subjectivity annotations. As shown in previous work, a high-precision classifier can be used to automatically generate subjectivity annotated data (Riloff and Wiebe, 2003). Additionally, the data annotated with a high-precision classifier can be used as a seed for bootstrapping methods, to further enrich each language individually.

We experiment with a majority vote meta-classifier, which combines the predictions of the *monolingual* Naïve Bayes classifiers described in Section 3. For a particular number of languages (one through six), all possible combinations of languages are considered. Each combination suggests a prediction only if its component classifiers agree, otherwise the system returns an ”unknown” prediction. The averages are computed across all the combinations featuring the same number of languages, regardless of language identity.

The results are shown in Table 4. The macro precision and recall averaged across groups formed using a given number of languages are presented in Figure 3. If the average monolingual classifier has a precision of 74.07%, the precision increases as more languages are considered, with a maximum precision of 83.38% obtained when the predictions of all six languages are considered (56.02% error reduction). It is interesting to note that the highest precision meta-classifier for groups of two languages includes German, while for groups with more than three languages, both Arabic and German are always present in the top performing combinations. English only appears in the highest precision combination for one, five and six languages, indicating the fact that the predictions based on Arabic and German are more robust.

We further analyze the behavior of each language considering only those meta-classifiers that include the given language. As seen in Figure 4, all languages experience a boost in performance

No lang	SubjP	SubjR	SubjF	ObjP	ObjR	ObjF	AllP	AllR	AllF
1	73.43%	82.76%	77.82%	74.70%	62.97%	68.33%	74.07%	72.86%	73.08%
2	76.88%	76.39%	76.63%	80.17%	54.35%	64.76%	78.53%	65.37%	70.69%
3	78.56%	72.42%	75.36%	82.58%	49.69%	62.02%	80.57%	61.05%	68.69%
4	79.61%	69.50%	74.21%	84.07%	46.54%	59.89%	81.84%	58.02%	67.05%
5	80.36%	67.17%	73.17%	85.09%	44.19%	58.16%	82.73%	55.68%	65.67%
6	**80.94**%	65.20%	72.23%	**85.83**%	42.32%	56.69%	**83.38**%	53.76%	64.46%

Table 4: Average measures for a particular number of languages in a combination (from one through six) for meta-classifiers

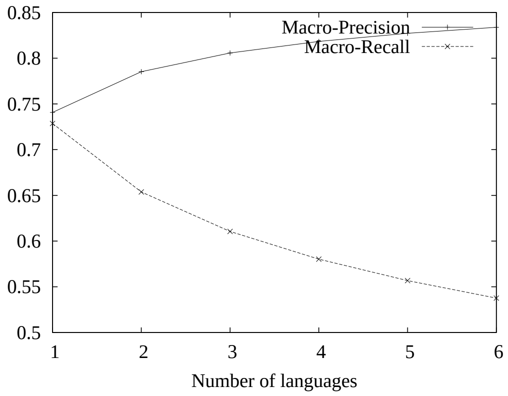


Figure 3: Average Macro-Precision and Recall across a given number of languages

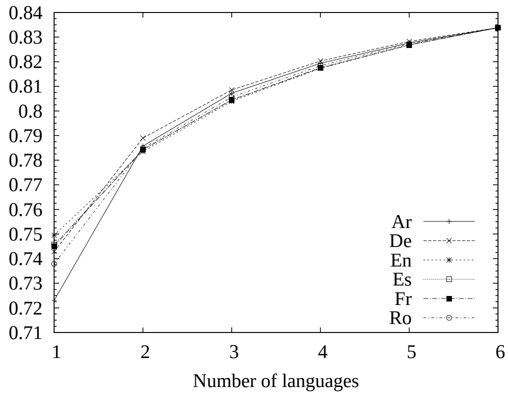


Figure 4: Average Macro-Precision relative to a given language

as a result of paired language reinforcement. Arabic gains an absolute 11.0% in average precision when considering votes from all languages, as compared to the 72.30% baseline consisting of the precision of the classifier using only monolingual features; this represents an error reduction in precision of 66.71%. The other languages experience a similar boost, including English which exhibits an error reduction of 50.75% compared to the baseline. Despite the fact that with each language that is added to the meta-classifier, the recall decreases, even when considering votes from all six languages, the recall is still reasonably high at 53.76%.

The results presented in table 4 are promising, as they are comparable to the ones obtained in previous work. Compared to (Wiebe et al., 2005), who used a high-precision rule-based classifier on the English MPQA corpus (see Table 1), our method has a precision smaller by 3.32%, but a recall larger by 21.16%. Additionally, unlike (Wiebe et al., 2005), which requires language-specific rules, making it applicable only to English, our method can be used to construct a high-precision classifier in any language that can be connected to English via machine translation.

6 Related Work

Recently, resources and tools for sentiment analysis developed for English have been used as a starting point to build resources in other languages, via cross-lingual projections or monolingual and multi-lingual bootstrapping. Several directions were followed, focused on leveraging annotation schemes, lexica, corpora and automated annotation systems. The English annotation scheme developed by (Wiebe et al., 2005) for opinionated text lays the groundwork for the research carried out by (Esuli et al., 2008) when annotating expressions of private state in the Italian Content Annotation Bank. Sentiment and subjectivity lexica such as the one included with the OpinionFinder distribution (Wiebe and Riloff, 2005), the General Inquirer (Stone et al., 1967), or the SentiWordNet (Esuli and Sebastiani, 2006b) were transfered into Chinese (Ku et al., 2006; Wu, 2008) and into Romanian (Mihalcea et al., 2007). English corpora manually annotated for subjectivity or sentiment such as MPQA (Wiebe et al., 2005), or the multi-domain sentiment classification corpus (Blitzer et al., 2007) were subjected to experiments in Spanish, Romanian, or Chinese upon automatic translation by (Banea et al., 2008; Wan, 2009). Furthermore, tools developed for English were used to determine sentiment or subjectivity labeling for a given target language by transferring the text to English and applying an English classifier on the resulting data. The labels were then transfered back into the target language (Bautin et al., 2008; Banea et al., 2008). These experiments are carried out in Arabic, Chinese, English, French, German, Italian, Japanese, Korean, Spanish, and Romanian.

The work closest to ours is the one proposed by (Wan, 2009), who constructs a polarity co-training system by using the multi-lingual views obtained through the automatic translation of product-reviews into Chinese and English. While this work proves that leveraging cross-lingual information improves sentiment analysis in Chinese over what could be achieved using monolingual resources alone, there are several major differences with respect to the approach we are proposing here. First, our training set is based solely on the automatic translation of the English corpus. We do not require an in-domain dataset available in the target language that would be needed for the co-training approach. Our method is therefore transferable to any language that has an English-to target language translation engine. Further, we focus on using multi-lingual data from six languages to show that the results are reliable and replicable across each language and that multiple languages aid not only in conducting subjectivity research in the target language, but also in improving the accuracy in the source language as well. Finally, while (Wan, 2009) research focuses on polarity detection based on reviews, our work seeks to determine sentence-level subjectivity from raw text.

7 Conclusion

Our results suggest that including multilingual information when modeling subjectivity can not only extrapolate current resources available for English into other languages, but can also improve subjectivity classification in the source language itself. We showed that we can improve an English classifier by using out-of-language features, thus achieving a 4.90% error reduction in accuracy with respect to using English alone. Moreover, we also showed that languages other than English can achieve an F-measure in subjectivity annotation of over 75%, without using any manually crafted resources for these languages. Furthermore, by combining the predictions made by monolingual classifiers using a majority vote learner, we are able to generate sentence-level subjectivity annotated data with a precision of 83% and a recall level above 50%. Such high-precision classifiers may be later used not only to create subjectivity-annotated data in the target language, but also to generate the seeds needed to sustain a language-specific bootstrapping.

To conclude and provide an answer to the question formulated in the title, more languages are better, as they are able to complement each other, and together they provide better classification results. When one language cannot provide sufficient information, another one can come to the rescue.

Acknowledgments

This material is based in part upon work supported by National Science Foundation awards #0917170 and #0916046. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Alm, Cecilia Ovesdotter, Dan Roth, and Richard Sproat. 1990. Emotions from text: machine learning for text-based emotion prediction. *Intelligence*.

Balog, Krisztian, Gilad Mishne, and Maarten De Rijke. 2006. Why Are They Excited? Identifying and Explaining Spikes in Blog Mood Levels. In *Proceedings of the*

11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-2006), Trento, Italy.

Banea, Carmen, Rada Mihalcea, Janyce Wiebe, and Samer Hassan. 2008. Multilingual Subjectivity Analysis Using Machine Translation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-2008)*, pages 127–135, Honolulu.

Bautin, Mikhail, Lohit Vijayarenu, and Steven Skiena. 2008. International Sentiment Analysis for News and Blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM-2008)*, Seattle, Washington.

Blitzer, John, Mark Dredze, and Fernando Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational (ACL-2007)*, pages 440–447, Prague, Czech Republic. Association for Computational Linguistics.

Carenini, Giuseppe, Raymond T Ng, and Xiaodong Zhou. 2008. Summarizing Emails with Conversational Cohesion and Subjectivity. In *Proceedings of the Association for Computational Linguistics: Human Language Technologies (ACL- HLT 2008)*, pages 353–361, Columbus, Ohio.

Esuli, Andrea and Fabrizio Sebastiani. 2006a. Determining Term Subjectivity and Term Orientation for Opinion Mining. In *Proceedings of the 11th Meeting of the European Chapter of the Association for Computational Linguistics (EACL-2006)*, volume 2, pages 193–200, Trento, Italy.

Esuli, Andrea and Fabrizio Sebastiani. 2006b. SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation*, pages 417–422.

Esuli, Andrea, Fabrizio Sebastiani, and Ilaria C Urciuoli. 2008. Annotating Expressions of Opinion and Emotion in the Italian Content Annotation Bank. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC-2008)*, Marrakech, Morocco.

Hu, Minqing and Bing Liu. 2004. Mining and Summarizing Customer Reviews. In *Proceedings of ACM Conference on Knowledge Discovery and Data Mining (ACM-SIGKDD-2004)*, pages 168–177, Seattle, Washington.

Ku, Lun-wei, Yu-ting Liang, and Hsin-hsi Chen. 2006. Opinion Extraction, Summarization and Tracking in News and Blog Corpora. In *Proceedings of AAAI-2006 Spring Symposium on Computational Approaches to Analyzing Weblogs*, number 2001, Boston, Massachusetts.

Lloyd, Levon, Dimitrios Kechagias, and Steven Skiena, 2005. *Lydia : A System for Large-Scale News Analysis (Extended Abstract) News Analysis with Lydia*, pages 161–166. Springer, Berlin / Heidelberg.

Mihalcea, Rada, Carmen Banea, and Janyce Wiebe. 2007. Learning Multilingual Subjective Language via Cross-Lingual Projections. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics (ACL-2007)*, pages 976–983, Prague, Czech Republic.

Riloff, Ellen and Janyce Wiebe. 2003. Learning Extraction Patterns for Subjective Expressions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 105–112, Sapporo, Japan.

Stone, Philip J, Marshall S Smith, Daniel M Ogilivie, and Dexter C Dumphy. 1967. *The General Inquirer: A Computer Approach to Content Analysis*. /. The MIT Press, 1st edition.

Wan, Xiaojun. 2009. Co-Training for Cross-Lingual Sentiment Classification. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*, Singapore.

Wiebe, Janyce and Rada Mihalcea. 2006. Word Sense and Subjectivity. In *Proceedings of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING-ACL-2006)*, Sydney, Australia.

Wiebe, Janyce and Ellen Riloff. 2005. Creating Subjective and Objective Sentence Classifiers from Unannotated Texts. In *Proceeding of CICLing-05, International Conference on Intelligent Text Processing and Computational Linguistics*, pages 486–497, Mexico City, Mexico.

Wiebe, Janyce, Theresa Wilson, and Claire Cardie. 2005. Annotating Expressions of Opinions and Emotions in Language. *Language Resources and Evaluation*, 39(2-3):165–210.

Wu, Yejun. 2008. Classifying attitude by topic aspect for English and Chinese document collections.

Yu, Hong and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentence. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 129–136, Sapporo, Japan.

Plagiarism Detection across Distant Language Pairs

Alberto Barrón-Cedeño **Paolo Rosso**
Natural Language Engineering Lab. - ELiRF
Universidad Politécnica de Valencia
{lbarron, prosso}@dsic.upv.es

Eneko Agirre **Gorka Labaka**
IXA NLP Group
Basque Country University
{e.agirre, gorka.labaka}@ehu.es

Abstract

Plagiarism, the unacknowledged reuse of text, does not end at language boundaries. Cross-language plagiarism occurs if a text is translated from a fragment written in a different language and no proper citation is provided. Regardless of the change of language, the contents and, in particular, the ideas remain the same. Whereas different methods for the detection of monolingual plagiarism have been developed, less attention has been paid to the cross-language case.

In this paper we compare two recently proposed cross-language plagiarism detection methods (CL-CNG, based on character n-grams and CL-ASA, based on statistical translation), to a novel approach to this problem, based on machine translation and monolingual similarity analysis (T+MA). We explore the effectiveness of the three approaches for less related languages. CL-CNG shows not be appropriate for this kind of language pairs, whereas T+MA performs better than the previously proposed models.

1 Introduction

Plagiarism is a problem in many scientific and cultural fields. Text plagiarism may imply different operations: from a simple cut-and-paste, to the insertion, deletion and substitution of words, up to an entire process of paraphrasing. Different models approach the detection of monolingual plagiarism (Shivakumar and García-Molina, 1995; Hoad and Zobel, 2003; Maurer et al., 2006). Each of these models is appropriate only in those cases where all the implied documents are written in the same language.

Nevertheless, the problem does not end at language boundaries. Plagiarism is also committed if the reused text is translated from a fragment written in a different language and no citation is provided. When plagiarism is generated by a translation process, it is known as cross-language plagiarism (CLP).

Less attention has been paid to the detection of this kind of plagiarism due to its enhanced difficulty (Ceska et al., 2008; Barrón-Cedeño et al., 2008; Potthast et al., 2010). In fact, in the recently held 1st International Competition on Plagiarism Detection (Potthast et al., 2009), no participants tried to approach it.

In order to describe the prototypical process of automatic plagiarism detection, we establish the following notation. Let d_q be a plagiarism suspect document. Let D be a representative collection of reference documents. D presumably includes the source of the potentially plagiarised fragments in d_q. Stein et al., (2007) divide the process into three stages[1]:

1. *heuristic retrieval of potential source documents*: given d_q, retrieving an appropriate number of its potential source documents $D^* \in D$ such that $|D^*| \lll |D|$;
2. *exhaustive comparison of texts*: comparing the text from d_q and $d \in D^*$ in order to identify reused fragments and their potential

[1] This schema was formerly proposed for monolingual plagiarism detection. Nevertheless, it can be applied without further modifications to the cross-language case.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 37–45,
Beijing, August 2010

sources; and

3. *knowledge-based post-processing*: those detected fragments with proper citation are discarded as they are not plagiarised.

The result is offered to the human expert to take the final decision. In the case of cross-language plagiarism detection (CLPD), the texts are written in different languages: $d_q \in L$ and $d' \in L'$.

In this research we focus on step 2: *cross-language exhaustive comparison of texts*, approaching it as an Information Retrieval problem of cross-language text similarity. Step 1, *heuristic retrieval*, may be approached by different CLIR techniques, such as those proposed by Dumais et al. (1997) and Pouliquen et al. (2003).

Cross-language similarity between texts, $\varphi(d_q, d')$, has been previously estimated on the basis of different models: multilingual thesauri (Steinberger et al., 2002; Ceska et al., 2008), comparable corpora —CL-Explicit Semantic Analysis CL-ESA— (Potthast et al., 2008), machine translation techniques —CL-Alignment-based Similarity Analysis CL-ASA— (Barrón-Cedeño et al., 2008; Pinto et al., 2009) and n-grams comparison —CL-Character n-Grams CL-CNG— (Mcnamee and Mayfield, 2004).

A comparison of CL-ASA, CL-ESA, and CL-CNG was carried out recently by Potthast et al. (2010). The authors report that in general, despite its simplicity, CL-CNG outperformed the other two models. Additionally, CL-ESA showed good results in the cross-language retrieval of topic-related texts, whereas CL-ASA obtained better results in exact (human) translations.

However, most of the language pairs used in the reported experiments (English-{German, Spanish, French, Dutch, Polish}) are related, whether because they have common predecessors or because a large proportion of their vocabularies share common roots. In fact, the lower syntactical relation between the English-Polish pair caused a performance degradation for CL-CNG, and for CL-ASA to a lesser extent. In order to confirm whether the closeness among languages is an important factor, this paper works with more distant language pairs: English-Basque and Spanish-Basque.

The rest of the paper is structured as follows. Section 2 describes the motivation for working on this research topic, stressing the situation of cross-language plagiarism among writers in less resourced languages. A brief overview of the few works on CLPD is included. The three similarity estimation models compared in this research work are presented in Section 3. The experimental framework and the obtained results are included in Section 4. Finally, Section 5 draws conclusions and discusses further work.

2 Motivation

Cases of CLP are common nowadays because information in multiple languages is available on the Web, but people still write in their own language. This special kind of plagiarism occurs more often when the target language is a less resourced one[2], as is the case of Basque.

Basque is a pre-indoeuropean language with less than a million speakers in the world and no known relatives in the language families (Wikipedia, 2010a). Still, Basque shares a portion of its vocabulary with its contact languages (Spanish and French). Therefore, we decided to work with two language pairs: Basque with Spanish, its contact language, and with English, perhaps the language with major influence over the rest of languages in the world. Although the considered pairs share most of their alphabet, the vocabulary and language typologies are very different. For instance Basque is an agglutinative language.

In order to illustrate the relations among these languages, Fig. 1 includes extracts from the English (*en*), Spanish (*es*) and Basque (*eu*) versions of the same Wikipedia article. The fragments are a sample of the lexical and syntactic distance between Basque and the other two languages. In fact, these sentences are completely co-derived and the corresponding entire articles are a sample of the typical imbalance in text available in the different languages (around $2,000$, $1,300$, and only

[2]Less resourced language is that with a low degree of representation on the Web (Alegria et al., 2009). Whereas the available text for German, French or Spanish is less than for English, the difference is more dramatic with other languages such as Basque.

The Party of European Socialists (PES) is a European political party comprising thirty-two socialist, social democratic and labour parties from each European Union member state and Norway.

El Partido Socialista Europeo (PSE) es un partido político pan-europeo cuyos miembros son de partidos socialdemócratas, socialistas y laboristas de estados miembros de la Unión Europea, así como de Noruega.

Europako Alderdi Sozialista Europar Batasuneko herrialdeetako eta Norvegiako hogeita hamahiru alderdi sozialista, sozialdemokrata eta laborista biltzen dituen alderdia da.

Figure 1: First sentences from the Wikipedia articles "Party of European Socialists" (*en*),"Partido Socialista Europeo" (*es*), and "Europako Alderdi Sozialista" (*eu*) (Wikipedia, 2010b).

100 words are contained in the *en*, *es* and *eu* articles, respectively).

Of high relevance is that the two corpora used in this work were manually constructed by translating English and Spanish text into Basque. In the experiments carried out by Potthast et al. (2010), which inspired our work, texts from the JCR-Acquis corpus (Steinberger et al., 2006) and Wikipedia were used. The first one is a multilingual corpus with no clear definition of source and target languages, whereas in Wikipedia no specific relationship exists between the different languages in which a topic may be broached. In some cases (cf. Fig. 1) they are clearly co-derived, but in others they are completely independent.

CLPD has been investigated just recently, mainly by adapting models formerly proposed for cross-language information retrieval. This is the case of cross-language explicit semantic analysis (CL-ESA), proposed by Potthast et al. (2008). In this case the comparison between texts is not carried out directly. Instead, a comparable corpus $C_{L,L'}$ is required, containing documents on multiple topics in the two implied languages. One of the biggest corpora of this nature is Wikipedia. The similarity between $d_q \in L$ and every document $c \in C_L$ is computed based on the cosine measure. The same process is made for L'. This step generates two vectors $[cos(d_q, c_1), \ldots, cos(d_q, c_{|C_L|})]$ and $[cos(d', c'_1), \ldots, cos(d', c'_{|C_{L'}|})]$, where each dimension is comparable between the two vectors. Therefore, the cosine between such vectors can be estimated in order to —indirectly— estimate how similar d_q and d' are. The authors suggest that this model can be used for CLPD.

Another recent model is *MLPlag*, proposed by Ceska et al. (2008). It exploits the *EuroWordNet Thesaurus*[3], that includes sets of synonyms in multiple European languages, with common identifiers across languages. The authors report experiments over a subset of documents of the English and Czech sections of the JRC-Acquis corpus as well as a corpus of simplified vocabulary[4]. The main difficulty they faced was the amount of words in the documents not included in the thesaurus (approximately 50% of the vocabulary).

This is a very similar approach to that proposed by Pouliquen et al. (2003) for the identification of document translations. In fact, both approaches have something in common: translations are searched at document level. It is assumed that an entire document has been reused (translated). Nevertheless, a writer is free to plagiarise text fragments from different sources, and compose a mixture of original and reused text.

A third model is the cross-language alignment-based similarity analysis (CL-ASA), proposed by Barrón-Cedeño et al. (2008), which is based on statistical machine translation technology. This model was proposed to detect plagiarised text fragments (similar models have been proposed for extraction of parallel sentences from comparable corpora (Munteanu et al., 2004)). The authors report experiments over a short set of texts from which simulated plagiarism was created from English to Spanish. Human as well as automatic machine translations were included in the collection. Further descriptions of this model are included in Section 3, as it is one of those being assessed in this research work.

To the best of our knowledge, no work (including the three previously mentioned) has been done considering less resourced languages. In this research work we approach the not uncommon problem of CLPD in Basque, with source texts written in Spanish (the co-official language of the

[3] `http://www.illc.uva.nl/EuroWordNet/`

[4] The authors do not mention the origin of the documents.

	low	tok	pd	bd	sd	lem
T+MA	■	■				■
CL-ASA	■	■				■
CL-CNG	■		■	■	■	

Table 1: Text preprocessing operations required for the different models. low=lowercasing, tok=tokenization, pd=punctuation marks deletion, bd=blank space deletion, sd=symbols deletion, lem=lematization.

Basque Country) and English (the language with most available texts in the world).

We compare three cross-language similarity analysis methods: T+MA (translation followed by monolingual analysis), a novel method based on machine translation followed by a monolingual similarity estimation; CL-CNG, a character n-gram based comparison model; and CL-ASA a model that combines translation and similarity estimation in a single step. Neither MLPlag nor CL-ESA are included in the comparison. On the one hand, we are interested in plagiarism at sentence level, and MLPlag is designed to compare entire documents. On the other hand, in previous experiments over exact translations, CL-ASA has shown to outperform it on language pairs whose alphabet or syntax are unrelated (Potthast et al., 2010). This is precisely the case of *en-eu* and *es-eu* language pairs. Additionally, the amount of Wikipedia articles in Basque available for the construction of the required comparable corpus is insufficient for the CL-ESA data requirements.

3 Definition of Models

In this section, we describe the three cross-language similarity models we compare. For experimental purposes (cf. Section 4) we consider d_q to be a suspicious sentence written in L and D' to be a collection of potential source sentences written in L' ($L \neq L'$). The text pre-processing required by the different models is summarised in Table 1. Examples illustrating how the models work are included in Section 4.3.

3.1 Translation + Monolingual Analysis

$d_q \in L$ is translated into L' on the basis of the Giza++ (Och and Ney, 2003), Moses (Koehn et al., 2007) and SRILM (Stolcke, 2002) tools, generating d'_q. The translation system uses a log-linear combination of state-of-the-art features, such as translation probabilities and lexical translation models on both directions and a target language model. After translation, d'_q and d' are lexically related, making possible a monolingual comparison.

Multiple translations from d_q into d'_q are possible. Therefore, performing a monolingual similarity analysis based on "traditional" techniques, such as those based on word n-grams comparison (Broder, 1997) or hash collisions (Schleimer et al., 2003), is not an option. Instead, we take the approach of the bag-of-words, which has shown good results in the estimation of monolingual text similarity (Barrón-Cedeño et al., 2009). Words in d'_q and d' are weighted by the standard $tf\text{-}idf$, and the similarity between them is estimated by the cosine similarity measure.

3.2 CL-Alignment-based Similarity Analysis

In this model an estimation of how likely is that d' is a translation of d_q is performed. It is based on the adaptation of the Bayes rule for MT:

$$p(d' \mid d_q) = \frac{p(d')\ p(d_q \mid d')}{p(d_q)}. \qquad (1)$$

As $p(d_q)$ does not depend on d', it is neglected. From an MT point of view, the conditional probability $p(d_q \mid d')$ is known as *translation model probability* and is computed on the basis of a statistical bilingual dictionary. $p(d')$ is known as *language model probability*; it describes the target language L' in order to obtain grammatically acceptable translations (Brown et al., 1993).

Translating d_q into L' is not the concern of this method, rather it focuses on retrieving texts written in L' which are potential translations of d_q. Therefore, Barrón-Cedeño et al. (2008) proposed replacing the language model (the one used in T+MA) by that known as *length model*. This model depends on text's character lengths instead of language structures.

Multiple translations from d into L' are possible, and it is uncommon to find a pair of translated texts d and d' such that $|d| = |d'|$. Nevertheless, the length of such translations is closely related to a translation length factor. In accordance with Pouliquen et al. (2003), the length model is defined as:

$$\varrho(d') = e^{-0.5\left(\frac{\frac{|d'|}{|d_q|} - \mu}{\sigma}\right)^2}, \quad (2)$$

where μ and σ are the mean and the standard deviation of the character lengths between translations of texts from L into L'. If the length of d' is not the expected given d_q, it receives a low qualification.

The translation model probability is defined as:

$$p(d \mid d') = \prod_{x \in d} \sum_{y \in d'} p(x, y), \quad (3)$$

where $p(x, y)$, a statistical bilingual dictionary, represents the likelihood that x is a valid translation of y. After estimating $p(x, y)$ from a parallel corpus, on the basis of the IBM statistical translation models (Brown et al., 1993), we consider, for each word x, only the k best translations y (those with the highest probabilities) up to a minimum probability mass of 0.4. This threshold was empirically selected as it eliminated noisy entries without discarding an important amount of relevant pairs.

The similarity estimation based on CL-ASA is finally computed as:

$$\varphi(d_q, d') = \varrho(d')\ p(d_q \mid d'). \quad (4)$$

3.3 CL-Character n-Gram Analysis

This model, the simplest of those compared in this research, has been used in (monolingual) Authorship Attribution (Keselj et al., 2003) as well as cross-language Information Retrieval (Mcnamee and Mayfield, 2004). The simplified alphabet considered is $\Sigma = \{a, \ldots, z, 0, \ldots, 9\}$; any other symbol is discarded (cf. Table 1). The resulting text strings are codified into character 3-grams, which are weighted by the standard tf-idf (considering this n has previously shown to produce the best results). The similarity between such representations of d_q and d' is estimated by the cosine similarity measure.

4 Experiments

The objective of our experiments is to compare the performance of the three similarity estimation models. Section 4.1 introduces the corpora we have exploited. The experimental framework is described in Section 4.2. Section 4.3 illustrates how the models work, and the obtained results are presented and discussed in Section 4.4.

4.1 Corpora

In other Information Retrieval tasks a plethora of corpora is available for experimental and comparison purposes. However, plagiarism implies an ethical infringement and, to the best of our knowledge, there is no corpora of actual cases available, other than some seminal efforts on creating corpora of text reuse (Clough et al., 2002), artificial plagiarism (Potthast et al., 2009), and simulated plagiarism (Clough and Stevenson, 2010). The problem is worse for cross-language plagiarism.

Therefore, in our experiments we use two parallel corpora: *Software*, an *en-eu* translation memory of software manuals generously supplied by Elhuyar Fundazioa[5]; and *Consumer*, a corpus extracted from a consumer oriented magazine that includes articles written in Spanish along with their Basque, Catalan, and Galician translations[6] (Alcázar, 2006). *Software* includes $288,000$ parallel sentences; 8.66 (6.83) words per sentence in the English (Basque) section. *Consumer* contains $58,202$ sentences; 19.77 (15.20) words per sentence in Spanish (Basque). These corpora also reflect the imbalance of text available in the different languages.

4.2 Experimental Framework

We consider D_q and D' to be two entire documents from which plagiarised sentences and their source are to be detected. We work at this level of granularity, and not entire documents, for two main reasons: (*i*) we are focused on the exhaustive comparison stage of the plagiarism detection process (cf. Section 1); and (*ii*) even a single sentence could be considered a case of plagiarism, as it transmits a complete idea. However, a plagiarised sentence is usually not enough to automatically negate the validity of an entire document. This decision is left to the human expert, which can examine the documents where several plagiarised sentences occur. Note that the task becomes computationally more expensive as, for every sentence, we are looking through thousands

[5] `http://www.elhuyar.org`
[6] `http://revista.consumer.es`

	es-eu		en-eu	
	μ	σ	μ	σ
f_1	1.1567	0.2346	1.0561	0.5497
f_2	1.1569	0.2349	1.0568	0.5510
f_3	1.1571	0.2349	1.0566	0.5433
f_4	1.1565	0.2363	1.0553	0.5352
f_5	1.1571	0.2348	1.0553	0.5467
$avg.$	1.1569	0.2351	1.0560	0.5452

Table 2: Length models estimated for each training partition $f_{1,\ldots,5}$. The values describe a normal distribution centred in $\mu \pm \sigma$, representing the expected length of the source text given the suspicious one.

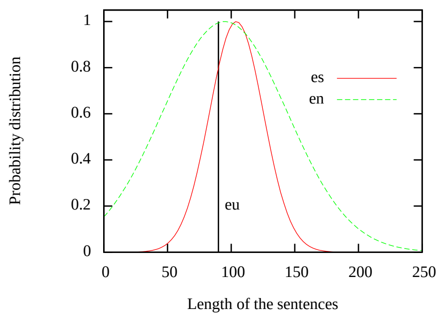


Figure 2: Example length factor for a sentence written in Basque (*eu*) d_q, such that $|d_q| = 90$. The normal distributions represent the expected lengths for the translation d', either in Spanish (*es*) or English (*en*).

of topically-related sentences that are potential sources of d_q, and not only those of a specific document.

CLPD is considered a ranking problem. Let $d_q \in D_q$ be a plagiarism suspicious sentence and $d' \in D'$ be its source sentence. We consider that the result of the process is correct if, given d_q, d' is properly retrieved. A 5-fold cross validation for both *en-eu* and *es-eu* was performed. Bilingual dictionaries, language and length models were estimated with the corresponding training partitions. The computed values for μ and σ are those included in Table 2. The values for the different partitions are very similar, showing the low variability in the translation lengths. On the basis of these estimated parameters, an example of length factor for a specific sentence is plotted in Fig. 2.

In the test partitions, for each suspicious sentence d_q, $11,640$ source candidate sentences exist for *es-eu* and $57,290$ for *en-eu*. This results in more than 135 million and 3 billion comparisons carried out for *es-eu* and *en-eu* respectively.

x_{eu}	y_{en}	$p(x,y)$	x_{eu}	y_{en}	$p(x,y)$
beste	another	0.288	**beste**	**other**	**0.348**
dokumentu	**document**	**0.681**	batzu	some	0.422
makro	**macro**	**0.558**	**ezin**	**not**	**0.179**
ezin	cannot	0.279	izan	is	0.241
izan	the	0.162	atzi	access	0.591
.	**.**	**0.981**			

Table 3: Entries in the bilingual dictionary for the words in d_q. Relevant entries for the example are in bold.

4.3 Illustration of Models

In order to clarify how the different models work, consider the following sentence pair, a suspicious sentence d_q written in Basque and its source d' written in English (sentences are short for illustrative purposes):

d_q *beste dokumentu batzuetako makroak ezin dira atzitu.*
d' *macros from other documents are not accessible.*

CL-CNG Example

In this case, symbols and spaces are discarded. Sentences become:

d_q *bestedokumentubatzuetakomakroakezindiraatzitu*
d' *macrosfromotherdocumentsarenotaccessible*

Only three 3-grams appear in both sentences (*ume*, *men*, *ent*). In order to keep the example simple, the 3-grams are weighted by tf only (in the actual experiments, tf-idf is used), resulting in a dot product of 3. The corresponding vectors magnitudes are $|\mathbf{d_q}| = 6.70$ and $|\mathbf{d}'| = 5.65$. Therefore, the estimated similarity is $\varphi(d_q, d') = 0.079$.

CL-ASA Example

In this case, the text must be tokenised and lemmatised, resulting in the following string:

d_q *beste dokumentu batzu makro ezin izan atzi .*
d' *macro from other document be not accessible .*

The sentences' lengths are $|d_q| = 38$ and $|d'| = 39$. Therefore, on the basis of Eq. 2, the length factor between them is $\varrho(d_q, d') = 0.998$.

The relevant entries of the previously estimated dictionary are included in Table 3. Such entries are substituted in Eq. 3, and the overall process results in a similarity $\varphi(d_q, d') = 2.74$. Whereas not a stochastic value, this is a weight used when ranking all the potential source sentences in D'.

T+MA Example

In this case, the same pre-processing than in CL-ASA is performed. In T+MA d_q is translated into L', resulting in the new pair:

d'_q *other document macro cannot be access .*
d' *macro from other document be not accessible .*

Note that d'_q is a valid translation of d_q. Nevertheless, it has few syntactic relation to d'. Therefore, applying more sophisticated codifications than the cosine measure over bag-of-words is not an option. The example is again simplified by weighting the words based on tf. Five words appear in both sentences, resulting in a dot product of 5. The vectors magnitudes are $|\mathbf{d'_q}| = |\mathbf{d'}| = \sqrt{7}$. The estimation by T+MA is $\varphi(d_q, d') = 0.71$, a high similarity level.

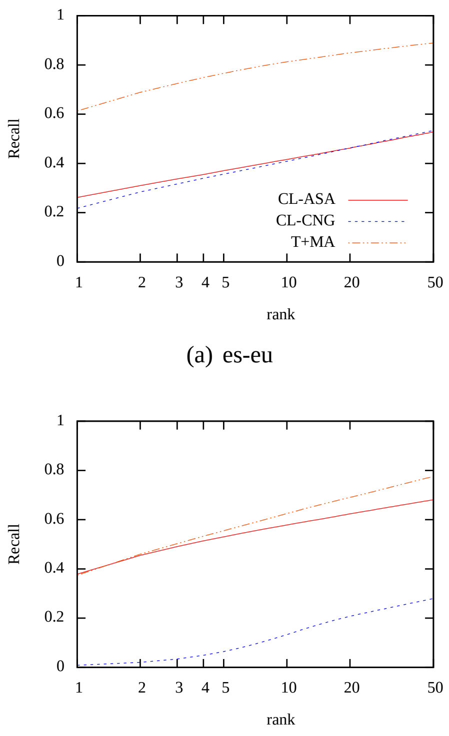
4.4 Results and Discussion

For evaluation we consider a standard measure: Recall. More specifically Recall after n texts have been retrieved ($n = [1 \ldots, 50]$). Figure 3 plots the average Recall value obtained in the 5-folds with respect to the rank position (n).

In both language pairs, CL-CNG obtained worse results than those reported for English-Polish by Potthast et al. (2010): $R@50 = 0.68$ vs. $R@50 = 0.53$ for *es-eu* and 0.28 for *en-eu*. This is due to the fact that neither the vocabulary nor its corresponding roots keep important relations. Therefore, when language pairs have a low syntactical relationship, CL-CNG is not an option. Still, CL-CNG performs better with *es-eu* than with *en-eu* because the first pair is composed of contact languages (cf. Section 1).

About CL-ASA, the results obtained with *es-eu* and *en-eu* are quite different: $R@50 = 0.68$ for *en-eu* and $R@50 = 0.53$ for *es-eu*. Whereas in the first case they are comparable to those of CL-CNG, in the second one CL-ASA completely outperforms it. The improvement of CL-ASA obtained for *en-eu* is due to the size of the training corpus available in this case (approximately five times the number of sentences available for *es-eu*). This shows the sensitivity of the model with respect to the size of the available resources.

Lastly, although T+MA is a simple approach that reduces the cross-language similarity estimation to a translation followed by a monolingual process, it obtained a good performance (R@50= 0.77 for *en-eu* and R@50=0.89 for es-eu). Moreover, this method proved to be less sensitive than CL-ASA to the lack of resources. This could be due to the fact that it considers both directions of the translation model (*e[n|s]-eu* and *eu-e[n|s]*). Additionally, the language model, applied in order to compose syntactically correct translations, reduces the amount of wrong translations and, indirectly, includes more syntactic information in the process. On the contrary, CL-ASA only considers one direction translation model *eu-e[n|s]* and completely disregards syntactical relations between the texts.



(a) es-eu

(b) en-eu

Figure 3: Evaluation of the cross-language ranking. Results plotted as rank versus Recall for the three evaluated models and the two language pairs ($R@[1, \ldots, 50]$).

Note that the better results come at the cost of higher computational demand. CL-CNG only requires easy to compute string comparisons. CL-ASA requires translation probabilities from aligned corpora, but once the probabilities are estimated, cross-language similarity can be computed very fast. T+MA requires the previous translation of all the texts, which can be very costly for large collections.

5 Conclusions and Further Work

In a society where information in multiple languages is available on the Web, cross-language plagiarism is occurring every day with increasing frequency. Still, cross-language plagiarism detection has not been approached sufficiently due to its intrinsic complexity. Though few attempts have been made, even less work has been made to tackle this problem for less resourced languages, and to explore distant language pairs.

We investigated the case of Basque, a language where, due to the lack of resources, cross-language plagiarism is often committed from texts in Spanish and English. Basque has no known relatives in the language family. However, it shares some of its vocabulary with Spanish.

Two state-of-the-art methods based on translation probabilities and n-gram overlapping, and a novel technique based on statistical machine translation were evaluated. The novel technique obtains the best results in both language pairs, with the n-gram overlap technique performing worst. In this sense, our results complement those of Potthast et al. (2010), which includes closely related language pairs as well.

Our results also show that better results come at the cost of more expensive processing time. For the future, we would like to investigate such performance trade-offs in more demanding datasets.

For future work we consider that exploring semantic text features across languages could improve the results. It could be interesting to further analyse how the reordering of words through translations might be relevant for this task. Additionally, working with languages even more distant from each other, such as Arabic or Hindi, seems to be a challenging and interesting task.

Acknowledgements

The research work of the first two authors is partially funded by CONACYT-Mexico and the MICINN project TEXT-ENTERPRISE 2.0 TIN2009-13391-C04-03 (Plan I+D+i). The research work of the last two authors is partially funded by the MICINN projects OPENMT-2 TIN2009-14675-C03-01 and KNOW2 TIN2009-14715-C04-01.

References

Alcázar, Asier. 2006. Towards Linguistically Searchable Text. In *Proceedings of the BIDE 2005*, Bilbao, Basque Country.

Alegria, Iñaki, Mikel L. Forcada, and Kepa Sarasola, editors. 2009. *Proceedings of the SEPLN 2009 Workshop on Information Retrieval and Information Extraction for Less Resourced Languages*, Donostia, Basque Country. University of the Basque Country.

Barrón-Cedeño, Alberto, Paolo Rosso, David Pinto, and Alfons Juan. 2008. On Cross-lingual Plagiarism Analysis Using a Statistical Model. In Stein, Stamatatos, and Koppel, editors, *ECAI 2008 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 2008)*, pages 9–13, Patras, Greece. CEUR-WS.org.

Barrón-Cedeño, Alberto, Andreas Eiselt, and Paolo Rosso. 2009. Monolingual Text Similarity Measures: A Comparison of Models over Wikipedia Articles Revisions. In Sharma, Verma, and Sangal, editors, *ICON 2009*, pages 29–38, Hyderabad, India. Macmillan Publishers.

Broder, Andrei Z. 1997. On the Resemblance and Containment of Documents. In *Compression and Complexity of Sequences (SEQUENCES'97)*, pages 21–29. IEEE Computer Society.

Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2):263–311.

Ceska, Zdenek, Michal Toman, and Karel Jezek. 2008. Multilingual Plagiarism Detection. In *Proceedings of the 13th International Conference on Artificial Intelligence*, pages 83–92. Springer Verlag Berlin Heidelberg.

Clough, Paul and Mark Stevenson. 2010. Developing a Corpus of Plagiarised Short Answers. *Language Resources and Evaluation: Special Issue on Plagiarism and Authorship Analysis*.

Clough, Paul, Robert Gaizauskas, and Scott Piao. 2002. Building and Annotating a Corpus for the Study of Journalistic Text Reuse. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC 2002)*, volume V, pages 1678–1691, Las Palmas, Spain.

Dumais, Susan T., Todd A. Letsche, Michael L. Littman, and Thomas K. Landauer. 1997. Automatic Cross-Language Retrieval Using Latent Semantic Indexing. In *AAAI-97 Spring Symposium Series: Cross-Language Text and Speech Retrieval*, pages 24—26. Stanford University.

Hoad, Timothy C. and Justin Zobel. 2003. Methods for Identifying Versioned and Plagiarized Documents. *Journal of the American Society for Information Science and Technology*, 54(3):203–215.

Keselj, Vlado, Fuchun Peng, Nick Cercone, and Calvin Thomas. 2003. N-gram-based Author Profiles for Authorship Attribution. In *Proceedings of the Conference Pacific Association for Computational Linguistics, PACLING'03*, pages 255–264, Halifax, Canada.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Annual Meeting of the Association for Computational Linguistics (ACL), demonstration session*, Prague, Czech Republic.

Maurer, Hermann, Frank Kappe, and Bilal Zaka. 2006. Plagiarism - A Survey. *Journal of Universal Computer Science*, 12(8):1050–1084.

Mcnamee, Paul and James Mayfield. 2004. Character N-Gram Tokenization for European Language Text Retrieval. *Information Retrieval*, 7(1-2):73–97.

Munteanu, Dragos S., Alexander Fraser, and Daniel Marcu. 2004. Improved Machine Translation Performace via Parallel Sentence Extraction from Comparable Corpora. In *Proceedings of the Human Language Technology and North American Association for Computational Linguistics Conference (HLT/NAACL 2004)*, Boston, MA.

Och, Frank Josef and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51. See also http://www.fjoch.com/GIZA++.html.

Pinto, David, Jorge Civera, Alberto Barrón-Cedeño, Alfons Juan, and Paolo Rosso. 2009. A Statistical Approach to Crosslingual Natural Language Tasks. *Journal of Algorithms*, 64(1):51–60.

Potthast, Martin, Benno Stein, and Maik Anderka. 2008. A Wikipedia-Based Multilingual Retrieval Model. In Macdonald, Ounis, Plachouras, Ruthven, and White, editors, *30th European Conference on IR Research, ECIR 2008, Glasgow*, volume 4956 LNCS of *Lecture Notes in Computer Science*, pages 522–530, Berlin Heidelberg New York. Springer.

Potthast, Martin, Benno Stein, Andreas Eiselt, Alberto Barrón-Cedeño, and Paolo Rosso. 2009. Overview of the 1st International Competition on Plagiarism Detection. In Stein, Rosso, Stamatatos, Koppel, and Agirre, editors, *SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (PAN 09)*, pages 1–9, San Sebastian, Spain. CEUS-WS.org.

Potthast, Martin, Alberto Barrón-Cedeño, Benno Stein, and Paolo Rosso. 2010. Cross-Language Plagiarism Detection. *Language Resources and Evaluation, Special Issue on Plagiarism and Authorship Analysis*.

Pouliquen, Bruno, Ralf Steinberger, and Camelia Ignat. 2003. Automatic Identification of Document Translations in Large Multilingual Document Collections. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP-2003)*, pages 401–408, Borovets, Bulgaria.

Schleimer, Saul, Daniel S. Wilkerson, and Alex Aiken. 2003. Winnowing: Local Algorithms for Document Fingerprinting. In *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, New York, NY. ACM.

Shivakumar, Narayanan and Hector García-Molina. 1995. SCAM: A Copy Detection Mechanism for Digital Documents. In *Proceedings of the 2nd Annual Conference on the Theory and Practice of Digital Libraries*.

Stein, Benno, Sven Meyer zu Eissen, and Martin Potthast. 2007. Strategies for Retrieving Plagiarized Documents. In Clarke, Fuhr, Kando, Kraaij, and de Vries, editors, *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 825–826, Amsterdam, The Netherlands. ACM.

Steinberger, Ralf, Bruno Pouliquen, and Johan Hagman. 2002. Cross-lingual Document Similarity Calculation Using the Multilingual Thesaurus EUROVOC. *Computational Linguistics and Intelligent Text Processing. Proceedings of the CICLing 2002*, 2276:415—424.

Steinberger, Ralf, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaz Erjavec, Dan Tufis, and Dániel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, volume 9, Genoa, Italy.

Stolcke, Andreas. 2002. SRILM - An Extensible Language Modeling toolkit. In *Intl. Conference on Spoken Language Processing*, Denver, Colorado.

Wikipedia. 2010a. Basque language. [Online; accessed 5-February-2010].

Wikipedia. 2010b. Party of European Socialists | Partido Socialista Europeo | Europako Alderdi Sozialista . [Online; accessed 10-February-2010].

Automatic Detection of Non-deverbal Event Nouns for Quick Lexicon Production

Núria Bel
IULA
Universitat Pompeu Fabra
nuria.bel@upf.edu

Maria Coll
IULA
Universitat Pompeu Fabra
maria.coll02@campus.upf.edu

Gabriela Resnik
Universidad Nacional de General Sarmiento and Universitat Pompeu Fabra
gresnik@ungs.edu.ar

Abstract

In this work we present the results of experimental work on the development of lexical class-based lexica by automatic means. Our purpose is to assess the use of linguistic lexical-class based information as a feature selection methodology for the use of classifiers in quick lexical development. The results show that the approach can help reduce the human effort required in the development of language resources significantly.

1 Introduction

Although language independent, many linguistic technologies are inherently tied to the availability of particular language data (i.e. Language Resources, LR). The nature of these data is very much dependent on particular technologies and the applications where are used. Currently, most systems are using LR collected by hand that still do not cover all languages, or all possible application domains, or all possible information required by the many applications that are being proposed. Methods for the automatic and quick development of new LR have to be developed in order to guarantee a supply of the required data. Esuli and Sebastiani (2006) did a classification experiment for creating lexica for opinion mining, for instance, and the importance of lexical information for event extraction in Biomedical texts has been addressed in Fillmore et al. (2006). One way of producing such resources is to classify words into lexical classes via methods based on their morphosyntactic contexts of occurrence.

In the next three sections we report on an experiment on cue-based lexical classification for non-deverbal event nouns, that is, nouns such as 'party' or 'conflict', which refer to an event but cannot be identified by their morphology, as is the case with deverbal nouns such as 'construction'. The purpose of this experiment was, as already stated, to investigate methods for the rapid generation of an event nouns lexicon for two different languages, using a reduced quantity of available texts. Assuming that linguistic information can be provided by occurrence distribution, as is usually done in linguistic theory to motivate lexical classes (e.g. Grimshaw, 1990), cue information has been gathered from texts and used to train and test a Decision Tree-based classifier. We experimented with two different languages to test the potential coverage of the proposed technique in terms of its adaptation to different languages, and also used different types of corpora to test its adaptability to different domains and sizes.

2 Some properties of Non-Deverbal Event Nouns in Spanish and English.

We based our experiment on the work by Resnik (2004) who proposes a specific lexical class for Spanish event nouns like *accidente* ('accident') or *guerra* ('war') which cannot be identified by suffixes such as '-ción' ('-tion') or 'miento' ('-ment'), i.e. the morphological marks of deverbal derivation. Her proposal of creating a new class is motivated by the syntactic behaviour of these non-deverbal event nouns that differ significantly both from deverbal nominalizations and from non event nouns. This proposal differs significantly from work such as Grimshaw (1990).

In Grimshaw (1990) a significant difference is shown to exist between process and result nominals, evident in certain ambiguous nouns such as *building*, which can have a process reading –

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 46–52,
Beijing, August 2010

in a sentence like *The building of the access road took three* weeks (= 'process of building')– and a non-eventive or result reading –in a sentence like *The building collapsed* (= 'edifice'). These two types of nominals differ in many lexico-syntactic properties, such as the obligatory/optional internal argument realization, the manner of external argument realization, the determiner selection and their ability to control infinitival clauses. Simple event nouns such as *trip* share several syntactic properties with result nominals, although their lexical meaning is indeed similar to that of the process or complex event nouns. The main difference is the fact that result nominals and simple event nouns, contrary to complex event nominals, are not verb-like in the way they combine with their satellites (Grimshaw 1990). The similarity between result nominals and simple event nouns is accepted in Picallo's (1991, 1999) analysis of Catalan and Spanish nominalizations and in Alexiadou's (2001) work on nominalizations in Greek, English, Hebrew and other languages.

Although the similarities between non-deverbal event nouns like *accidente* and result nominals are undeniable, some evidence (Resnik, 2004 and 2009) has been found that non-deverbal event nouns cannot be assimilated to either result nominals or simple non event nouns like *tren* ('train'), in spite of their shared properties. In the next sections, we briefly present evidence that non-deverbal event nouns are a separate lexical class and that this evidence can be used for identifying the members of this class automatically, both in Spanish and in English. Our hypothesis is that whenever there is a lexical class motivated by a particular distributional behaviour, a learner can be trained to identify the members of this class. However, there are two main problems to lexical classification: noise and silence, as we will see in section 4.

Resnik (2004) shows that non-deverbal event nouns occur in a unique combination of syntactic patterns: they are basically similar to result nouns (and simple non event nouns) regarding the realization of argument structure, yet they pattern along process nominals regarding event structure, given that they accept the same range of aspectual adjuncts and quantifiers as these nouns and are selected as subjects by the same 'aspectual' verbs (*empezar*, 'to start'; *durar*, 'to last', etc.) (cf. section 3.2). As to other nominal properties, such as the mass/count distinction, the contexts show that non-deverbal event nouns are not quite like either of the two kinds of nominalizations, and they behave like simple non event nouns. The table below summarizes the lexico-syntactic properties of the different nouns described by Grimshaw (1990) with the addition of Resnik's proposed new one.

	NDV E N (*war*)	PR-N (*construction* = event)	RES-N (*construction* = result. obj.)	NEN (map)
Obligatory internal argument	no	yes	no	No
External argument realization	genitive DP	PP_by	genitive DP	genitive DP
Subject of aspectual verbs (*begin, last..*)	yes	yes	no	no
Aspectual quantifier (*a period of*)	yes	yes	no	no
Complement of *during, ...*	yes	yes	no	no
Count/mass (determiners, plural forms)	mass/count	mass	count	mass/ count

Table 1. Lexico-syntactic properties of English Non-Deverbal Event Nouns (NDV E N), Process Nouns (PR-N) and Result Nouns (RES-N) and Non Event Nouns (NEN).

3 Automatic Detection of Non-deverbal Event Nouns

We have referred to the singularities of non-deverbal event nouns as a lexical class in contrast with other event and non-event nouns. In our experiment, we have extracted the characteristics of the contexts where we hypothesize that members of this class occur and we have used them as variables to train an automatic learner that can rely on these features to automatically classify words into those which are indeed non-deverbal event nouns and those which are not. Because deverbal result nouns are easily identifiable by the nominal suffix they bear (for instance, '-tion' for English and '-ción' for Spanish), our experiment has been centered in separating non-deverbal event nouns like *guerra/war* from non event nouns like *tren/train*.

Some work related to our experiments can be found in the literature dealing with the identification of new events for broadcast news and semantic annotation of texts, which are two possible applications of automatic event detection (Allan et al. 1998 and Saurí et al. 2005, respectively, for example). For these systems, however, it would be difficult to find non-deverbal event nouns because of the absence of morphological suffixes, and therefore they could benefit from our learner.

3.1 Cue-based Lexical Information Acquisition

According to the linguistic tradition, words that can be inserted in the same contexts can be said to belong to the same class. Thus, lexical classes are linguistic generalizations drawn from the characteristics of the contexts where a number of words tend to appear. Consequently, one of the approaches to lexical acquisition proposes to classify words taking as input characteristics of the contexts where words of the same class occur. The idea behind this is that differences in the distribution of the contexts will separate words in different classes, e.g. the class of transitive verbs will show up in passive constructions, while the intransitive verbs will not. Thus, the whole set of occurrences (tokens) of a word are taken as cues for defining its class (the class of the type), either because the word is observed in a number of particular contexts or because it is not. Selected references for this approach are: Brent, 1993; Merlo and Stevenson, 2001; Baldwin and Bond, 2003; Baldwin, 2005; Joanis and Stevenson, 2003; Joanis et al. 2007.

Different supervised Machine Learning (ML) techniques have been applied to cue-based lexical acquisition. A learner is supplied with classified examples of words represented by numerical information about matched and not matched cues. The final exercise is to confirm that the data characterized by the linguistically motivated cues support indeed the division into the proposed classes. This was the approach taken by Merlo and Stevenson (2001), who worked with a Decision Tree and selected linguistic cues to classify English verbs into three classes: *unaccusative*, *unergative* and *object-drop*. Animacy of the subject, for instance, is a significant cue for the class of object dropping verbs, in contrast with verbs in *unergative* and *unaccusative* classes. Baldwin and Bond (2003) used a number of linguistic cues (i.e. co-occurence with particular determiners, number, etc.) to learn the countability of English nouns. Bel et al. (2007) proposed a number of cues for classifying nouns into different types according to a lexical typology. The need for using more general cues has also been pointed out, such as the part of speech tags of neighboring words (Baldwin, 2005), or general linguistic information as in Joanis et al. (2007), who used the frequency of filled syntactic positions or slots, tense and voice features, etc., to describe the whole system of English verbal classes.

3.2 Cues for the Detection of Non-deverbal Event Nouns in Spanish

As we have seen in section 2, non-deverbal event nouns can be identified by their occurrence in particular syntactic and lexical contexts of co-occurrence.We have used 11 cues for separating non-deverbal event nouns from non event nouns in Spanish. These cues are the following:

Cues 1-3. Nouns occurring in PPs headed by prepositions such as *durante* ('during'), *hasta el final de* ('until the end of'), *desde el principio de* ('from the beginning of'), and similar expressions are considered to be eventive. Thus, occurrence after one of such expressions will be indicative of an event noun.

Cues 4-8. Nouns occurring as external or internal arguments of verbs such as *ocurrir* ('occur'), *producir* ('produce' or 'occur', in the case of ergative variant *producirse*), *celebrar* ('celebrate'), and others with similar meanings, are also events. Note that we identify as 'external arguments' the nouns occurring immediately after the verb in particular constructions, as our *pos*- tagged text does not contain information about subjects (see below). In many cases it is the internal argument occurring in these contexts. These verbs tend to appear in 'presentative' constructions such as *Se produjo un accidente* ('An accident occurred'), with the pronoun *se* signalling the lack of external argument. Verbs like *ocurrir* appear in participial absolute constructions or with participial adjectives, which means they are unaccusatives.

Cue 9. The presence of temporal quantifying expressions such as *dos semanas de* ('two weeks of') or similar would indicate the eventive character of a noun occurring with it, as mentioned in section 2.

Cue 10. Non-deverbal event nouns will not be in Prepositional Phrases headed by locative prepositions such as *encima de* ('on top of') or *debajo de* ('under'). These cues are used as negative evidence for non-event deverbal nouns.

Cue 11. Non-deverbal event nouns do have an external argument that can also be realized as an adjective. The alternation of DP arguments with adjectives was then a good cue for detecting non-deverbal events, even when some other nouns may appear in this context as well. For instance: *fiesta nacional* ('national party') vs. *mapa nacional* ('national map').

3.3 Cues for the Detection of Non-Deverbal Event Nouns in English

As for Spanish, cues for English were meant to separate the newly proposed class of non-deverbal event nouns from non-event nouns if such a class exists as well.

Cues 1-3. Process nominals and non-deverbal event nouns can be identified by appearing as complements of aspectual PPs headed by prepositions like *during*, *after* and *before*, and complex prepositions such as *at the end of* and *at the beginning of*.

Cues 4 and 5. Non-deverbal nouns may occur as external or internal arguments of aspectual as well as occurrence verbs such as *initiate*, *take place*, *happen*, *begin*, and *occur*. Those arguments are identified either as subjects of active or passive sentences, depending on the verb, i.e. *the therapy was initiated* and *the conflict took place*.

Cue 6. Likewise, nouns occurring in expressions such as *frequency of*, *occurrence of* and *period of* would probably be event nouns, i.e. *the frequency of droughts*.

Cue 7 and 8. Event nouns may as well appear as objects of aspectual and time-related verbs, such as in *have begun a campaign* or *have carried out a campaign*.

Cues 10 and 11. They are intended to register event nouns whose external argument, although optional, is realized as a genitive complement, e.g. *enzyme's loss*, even though this cue is shared with other types of nouns. Following the characterization suggested for Spanish, we also tried external arguments realized as adjectives in cue 11, as in *Napoleonic war*, but we found empirical evidence that it is not useful.

Cues 12-16. Finally, as in the experiment for Spanish, we have also included evidence that is more common for non-event nouns, that is, we have used negative evidence to tackle the problem of sparse data or silence discussed in the next section. It is considered a negative cue for a noun to be preceded by an indefinite determiner, to be in a PP headed by a locative preposition, and to be followed by the prepositions *by* or *of*, as a PP headed by one these prepositions could be an external argument and, as it has been noted above, the external argument of event nouns tends to be realized as a genitive complement (as in *John's trip/party*).

In the selection of these cues, we have concentrated on those that separate the class of non-deverbal event nouns from the class formed by simple non event nouns like *train*, where no particular deverbal suffix can assist their detection. If it is the case that these are really cues for detecting non-deverbal event nouns, the learner should confirm it by classifying non-deverbal event nouns correctly, separating them from other types of nouns.

4 Experiment and results

For our experiments we have used Regular Expressions to implement the patterns just mentioned, which look for the intended cues in a part-of-speech tagged corpus. We have used a corpus of 21M tokens from two Spanish newspapers (*El País* and *La Vanguardia*), and an English technical corpus made of texts dealing with varying subject matter (Economy, Medicine, Computer science and Environmental issues), of about 3.2M tokens. Both Spanish and English corpora are part of the Technical Corpus of IULA at the UPF (CT-IULA, Cabré et al. 2006). The positive or negative results of the n-pattern checking in all the occurrences of a word are stored in an n-dimension vector. Thus, a single vector summarizes all the occurrences of a word (the type) by encoding how many times each cue has been observed. Zero values, i.e. no matching, are also registered.

We used a Decision Tree (DT) classifier in the Weka (Witten and Frank, 2005) implementation of pruned C4.5 decision tree (Quinlan, 1993). The DT performs a general to specific search in a feature space, selecting the most informative attributes for a tree structure as the search proceeds. The goal is to select the minimal set of attributes that efficiently partitions the feature space into classes of observations and assemble them into a tree. During the experiment, we tuned the list of cues actually used in the classification task, because some of them turned out to be useless, as they did not show up even once in the corpus. This was especially true for the English corpus with cues 5, 11 and 12. Note that the English corpus is only 3.2 million words.

In the experiment we used a 10-fold cross-validation testing using manually annotated gold-standard files made of 99 non-event and 100 non-deverbal event nouns for Spanish and 93 non event and 74 non-deverbal event nouns for English[1]. In this first experiment, we decided to use mostly non-deverbal non event nouns such as *map*, because detecting result nouns like *construction* is easy enough, due to the deverbal suffix. However, for the English experiment, and because of the scarcity of non-deverbal nouns occurrences, we had to randomly select some deverbals that were not recognized by the suffix.

The results of our experiment gave a total accuracy of 80% for Spanish and 79.6% for English, which leads to think that corpus size is not a determinant factor and that this method can be used for addressing different languages, provided a good characterization of the lexical class in terms of particular occurrence distributions is achieved. Yet, although the accuracy of both English and Spanish test sets is similar, we will see later on that the size of the corpus does indeed affect the results.

An analysis of the errors shows that they can be classified in two groups: errors due to noise, and errors due to silence.

(i) Noise. In his seminal work, Brent (1993) already pointed out that "the cues occur in contexts that were not aimed at". Noise can be due to errors in processing the text, because we had only used low-level analysis tools. For instance, in "during the first world war" our RE cannot detect that "world" is not the head of the Noun Phrase. Brent's hypothesis, followed by most authors afterwards, is that noise can be eliminated by statistical methods because of its low frequency. However, the fact is that in our test set significant information is as sparse as noise, and the DT cannot correctly handle this. In our data sets, most of the false positives are due to noise.

(ii) Silence. Some nouns appear only once or twice in the corpus and do not show up in any of the sought contexts (for instance, *terremoto*, 'earthquake', in Spanish press). Moreover, this is independent of the size of the corpus, because the Zipfian distribution of tokens allows us to predict that there will always be low-frequency nouns. Low frequency words produce non informative vectors, with only zero-valued cues, and our classifier tends to classify non-informative vectors as non-event nouns, because most of the cues have been issued to identify event nouns. This was the main reason to introduce negative contexts as well as positive ones, as we mentioned in section 3.

However, these systematic sources of error can be taken as an advantage when assessing the usability of the resulting resources. Having about 80% of accuracy would not be enough to ensure the proper functioning of the application in which the resource is going to be used. So, in order to gain precision, we decided to separate the set of words that could be safely taken as correctly classified. Thus, we had used the confidence, i.e. probability of the classification de-

[1] **Positive:** accident, assembly, audience, battle, boycott, campaign, catastrophe, ceremony, cold, collapse, conference, conflict, course, crime, crisis, cycle, cyclone, change, choice, decline, disease, disaster, drought, earthquake, epidemic, event, excursion, fair, famine, feast, festival, fever, fight, fire, flight, flood, growth, holiday, hurricane, impact, incident, increase, injury, interview, journey, lecture, loss, meal, measurement, meiosis, marriage, mitosis, monsoon, period, process, program, quake, response, seminar, snowstorm, speech, storm, strike, struggle, summit, symposium, therapy, tour, treaty, trial, trip, vacation, war. **Negative:** agency, airport, animal, architecture, bag, battery, bird, bridge, bus, canal, circle, city, climate, community, company, computer, constitution, country, creature, customer, chain, chair, channel, characteristic, child, defence, director, drug, economy, ecosystem, energy, face, family, firm, folder, food, grade, grant, group, health, hope, hospital, house, illusion, information, intelligence, internet, island, malaria, mammal, map, market, mountain, nation, nature, ocean, office, organism, pencil, people, perspective, phone, pipe, plan, plant, profile, profit, reserve, river, role, satellite, school, sea, shape, source, space, star, statistics, store, technology, television, temperature, theme, theory, tree, medicine, tube, university, visa, visitor, water, weather, window, world.

cisions to assess which are below a reasonable level of confidence.

In the Spanish test set, for instance, precision of the positive classification, i.e. the percentage of words correctly classified as event nouns, raises from 0.82 to 0.95 when only instances of classification with a confidence of more than 0.8 are selected. In the figure below, we can see the precision curve for the Spanish test set.

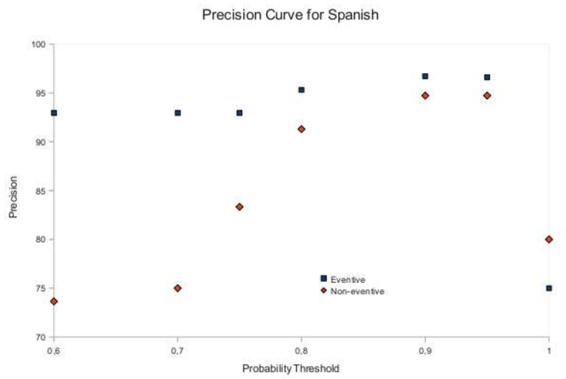


Figure 1: Precision curve for the Spanish test set.

In general, precision is higher when confidence is higher, except for complete confidence, 1, as we will explain later with the English case. This general behavior could be interpreted as a guarantee that there is a significant number of classified nouns (87 out of 199 for the Spanish test set with a threshold of 0.8 confidence) that need not to be manually reviewed, i.e. a 43% of the automatically acquired lexica can safely be considered correct. From figure 1, we can also see that the classifier is consistently identifying the class of non-deverbal event nouns even with a lower threshold. However, the resulting non-event noun set contains a significant number of errors. From the point of view of the usability, we could also say that only those words that are classified as non-event nouns must be revised.

Figure 2 for English test set shows a different behavior, which can only be justified because of the difference in corpus size. A small corpus increases the significance of silence errors. Fewer examples give less information to the classifier, which still makes the right decisions but with less confidence in general. However, for the extreme cases, for instance the case of 7 word vectors with only zero-values, the confidence is very high, that is 1, but the decisions are wrong. These cases of mostly zero values are wrongly considered to be non-events. This is the reason for the low precision of very confident decisions in English, i.e. sparse data and its consequence, silence.

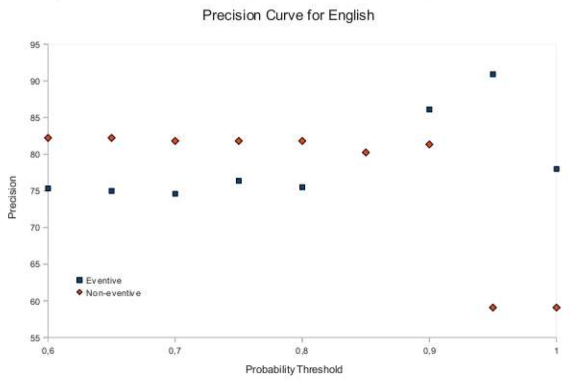


Figure 2: Precision curve for the English test set.

5 Conclusions

In this paper we have proposed the use of lexical classification methods based on differences in the distributional behavior of word classes for the quick production of lexica containing the information required by particular applications. We have dealt with non-deverbal event nouns, which cannot be easily recognized by any suffixes, and we have carried out a classification experiment, which consisted in training a DT with the information used in the linguistic literature to justify the existence of this class. The results of the classifier, close to 80% accuracy in two different languages and with different size and types of source corpora, show the validity of this very simple approach, which can be decisive in the production of lexica with the knowledge required by different technologies and applications in a time-efficient way. From the point of view of usability, this approach can be said to reduce the amount of work in more than a 40%.

Acknowledgements

We want to thank Muntsa Padró for her valuable contribution. This work was partially supported by the PANACEA project (EU-7FP-ITC-248064).

References

Alexiadou, A. (2001). Functional Structure in Nominals: Nominalization and Ergativity. Amsterdam/Philadelphia: John Benjamins Publishing Company.

Allan, J.; Papka, R.; Lavrenko, V. (1998). On-line New Event Detection and Tracking. SIGIR98, Melbourne, Australia.

Baldwin, T. and F. Bond. 2003. "Learning the Countability of English Nouns from Corpus Data". *Proceedings of the 41st. Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan.

Baldwin, T. 2005. General-Purpose Lexical Acquisition: Procedures, Questions and Results, In *Proceedings of the Pacific Association for Computational Linguistics 2005*, Tokyo, Japan

Bel, N.; Espeja, S.; Marimon, M. 2007. Automatic Acquisition of Grammatical Types for Nouns. In *H LT 2007: The Conference of the NAACL*. Companion Volume, Short Papers. Rochester, USA.

Brent, M. R. 1993, 'From grammar to lexicon: unsupervised learning of lexical syntax'. *Computational Linguistics* 19: 243-262.

Cabré, M. T.; Bach, C.; Vivaldi, J. 2006. *10 anys del Corpus de l'IULA*. Barcelona: Institut Universitari de Lingüística Aplicada. Universitat Pompeu Fabra

Esuli, A. and Sebastiani, F.. 2006. Determining term subjectivity and term orientation for opinion mining. In Proceedings of EACL-06, 11th Conference of the European Chapter of the Association for Computational Linguistics, Trento, IT.

Fillmore, Charles J.Srini Narayanan, and Collin F. Baker. 2006. What Can Linguistics Contribute to Event Extraction? *Proceedings of the 2006 AAAI Workshop on Event Extraction and Synthesis, page 18--23.*

Grimshaw, J. (1990). Argument Structure. Cambridge: The MIT Press.

Joanis, E; Stevenson, S; and James, D. 2007. A General Feature Space for Automatic Verb Classification. *Natural Language Engineering, 14*.

Korhonen, A. 2002. 'Subcategorization acquisition'. As Technical Report UCAM-CL-TR-530, University of Cambridge, UK.

Merlo P. and Stevenson S. 2001. Automatic Verb Classification based on Statistical Distribution of Argument Structure, *Computational Linguistics*, 27:3.

Picallo, M. C. (1999). "La estructura del sintagma nominal: Las nominalizaciones y otros sustantivos con complementos argumentales", en Bosque, I. & V. Demonte (eds.) *Gramática descriptiva de la lengua española*. Madrid: Real Academia Española / Espasa Calpe. Vol. 1, Cap. 6, 363-394.

Quinlan, R.J. 1993. C4.5: Programs for Machine Learning. Series in Machine Learning. Morgan Kaufman, San Mateo, CA.

Resnik, G. (2004). *Los nombres eventivos no deverbales en español*. Proyecto de Tesis. Institut Universitari de Lingüística Aplicada, Universitat Pompeu Fabra.

Resnik, G. (2009) "La determinación de la eventividad nominal en español". En G. Ciapuscio (ed.) *De la palabra al texto: estudios lingüísticos del español*. Buenos Aires: Eudeba.

Saurí, R.; R. Knippen, M. Verhagen and J. Pustejovsky. 2005. Evita: A Robust Event Recognizer for QA Systems. *Proceedings of HLT/EMNLP 2005:* 700-70

Witten, I. H. and Frank E. 2005. Data Mining: Practical machine learning tools and techniques. 2nd Edition, Morgan Kaufmann, San Francisco.

Fast and Accurate Arc Filtering for Dependency Parsing

Shane Bergsma
Department of Computing Science
University of Alberta
sbergsma@ualberta.ca

Colin Cherry
Institute for Information Technology
National Research Council Canada
colin.cherry@nrc-cnrc.gc.ca

Abstract

We propose a series of learned arc filters to speed up graph-based dependency parsing. A cascade of filters identify implausible head-modifier pairs, with time complexity that is first linear, and then quadratic in the length of the sentence. The linear filters reliably predict, in context, words that are roots or leaves of dependency trees, and words that are likely to have heads on their left or right. We use this information to quickly prune arcs from the dependency graph. More than 78% of total arcs are pruned while retaining 99.5% of the true dependencies. These filters improve the speed of two state-of-the-art dependency parsers, with low overhead and negligible loss in accuracy.

1 Introduction

Dependency parsing finds direct syntactic relationships between words by connecting head-modifier pairs into a tree structure. Dependency information is useful for a wealth of natural language processing tasks, including question answering (Wang et al., 2007), semantic parsing (Poon and Domingos, 2009), and machine translation (Galley and Manning, 2009).

We propose and test a series of **arc filters** for graph-based dependency parsers, which rule out potential head-modifier pairs before parsing begins. In doing so, we hope to eliminate implausible links early, saving the costs associated with them, and speeding up parsing. In addition to the scaling benefits that come with faster processing, we hope to enable richer features for parsing by constraining the set of arcs that need to be considered. This could allow extremely large feature sets (Koo et al., 2008), or the look-up of expensive corpus-based features such as word-pair mutual information (Wang et al., 2006). These filters could also facilitate expensive learning algorithms, such as semi-supervised approaches (Wang et al., 2008).

We propose three levels of filtering, which are applied in a sequence of increasing complexity:

Rules: A simple set of machine-learned rules based only on parts-of-speech. They prune over 25% of potential arcs with almost no loss in coverage. Rules save on the wasted effort for assessing implausible arcs such as DT $\rightarrow$ DT.

Linear: A series of classifiers that tag words according to their possible roles in the dependency tree. By treating each word independently and ensuring constant-time feature extraction, they operate in linear time. We view these as a dependency-parsing analogue to the span-pruning proposed by Roark and Hollingshead (2008). Our fast linear filters prune 54.2% of potential arcs while recovering 99.7% of true pairs.

Quadratic: A final stage that looks at pairs of words to prune unlikely arcs from the dependency tree. By employing a light-weight feature set, this high-precision filter can enable more expensive processing on the remaining plausible dependencies.

Collectively, we show that more than 78% of total arcs can be pruned while retaining 99.5% of the true dependencies. We test the impact of these filters at both train and test time, using two state-of-the-art discriminative parsers, demonstrating speed-ups of between 1.9 and 5.6, with little impact on parsing accuracy.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 53–61,
Beijing, August 2010

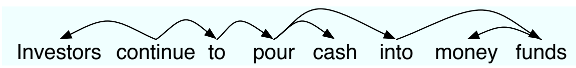


Figure 1: An example dependency parse.

2 Dependency Parsing

A dependency tree represents the syntactic structure of a sentence as a directed graph (Figure 1), with a node for each word, and arcs indicating head-modifier pairs (Meĺčuk, 1987). Though dependencies can be extracted from many formalisms, there is a growing interest in predicting dependency trees directly. To that end, there are two dominant approaches: graph-based methods, characterized by arc features in an exhaustive search, and transition-based methods, characterized by operational features in a greedy search (McDonald and Nivre, 2007). We focus on graph-based parsing, as its exhaustive search has the most to gain from our filters.

Graph-based dependency parsing finds the highest-scoring tree according to a scoring function that decomposes under an exhaustive search (McDonald et al., 2005). The most natural decomposition scores individual arcs, represented as head-modifier pairs $[h, m]$. This enables search by either minimum spanning tree (West, 2001) or by Eisner's (1996) projective parser. This paper focuses on the projective case, though our techniques transfer to spanning tree parsing. With a linear scoring function, the parser solves:

$$\mathrm{parse}(s) = \mathrm{argmax}_{t \in s} \sum_{[h,m] \in t} \bar{w} \cdot \bar{f}(h, m, s)$$

The weights $\bar{w}$ are typically learned using an online method, such as an averaged perceptron (Collins, 2002) or MIRA (Crammer and Singer, 2003). 2nd-order searches, which consider two siblings at a time, are available with no increase in asymptotic complexity (McDonald and Pereira, 2006; Carreras, 2007).

The complexity of graph-based parsing is bounded by two processes: parsing (carrying out the argmax) and arc scoring (calculating $\bar{w} \cdot \bar{f}(h, m, s)$). For a sentence with n words, projective parsing takes $O(n^3)$ time, while the spanning tree algorithm is $O(n^2)$. Both parsers require scores for arcs connecting each possible $[h, m]$ pair in s; therefore, the cost of arc scoring is also $O(n^2)$, and may become $O(n^3)$ if the features include words in s between h and m (Galley and Manning, 2009). Arc scoring also has a significant constant term: the number of features extracted for an $[h, m]$ pair. Our in-house graph-based parser collects on average 62 features for each potential arc, a number larger than the length of most sentences. With the cluster-based features suggested by Koo et al. (2008), this could easily grow by a factor of 3 or 4.

The high cost of arc scoring, coupled with the parsing stage's low grammar constant, means that graph-based parsers spend much of their time scoring potential arcs. Johnson (2007) reports that when arc scores have been precomputed, the dynamic programming component of his 1st-order parser can process an amazing 3,580 sentences per second.[1] Beyond reducing the number of features, the easiest way to reduce the computational burden of arc scoring is to score only plausible arcs.

3 Related Work

3.1 Vine Parsing

Filtering dependency arcs has been explored primarily in the form of vine parsing (Eisner and Smith, 2005; Dreyer et al., 2006). Vine parsing establishes that, since most dependencies are short, one can parse quickly by placing a hard constraint on arc length. As this coarse filter quickly degrades the best achievable performance, Eisner and Smith (2005) also consider conditioning the constraint on the part-of-speech (PoS) tags being linked and the direction of the arc, resulting in a separate threshold for each $[\mathrm{tag}(h), \mathrm{tag}(m), \mathrm{dir}(h, m)]$ triple. They sketch an algorithm where the thresholded length for each triple starts at the highest value seen in the training data. Thresholds are then decreased in a greedy fashion, with each step producing the smallest possible reduction in reachable training arcs. We employ this algorithm as a baseline in our experiments. To our knowledge, vine parsing

[1] To calibrate this speed, consider that the publicly available 1st-order MST parser processes 16 sentences per second on modern hardware. This includes I/O costs in addition to the costs of arc scoring and parsing.

has not previously been tested with a state-of-the-art, discriminative dependency parser.

3.2 CFG Cell Classification

Roark and Hollingshead (2008) speed up another exhaustive parsing algorithm, the CKY parser for CFGs, by classifying each word in the sentence according to whether it can open (or close) a multi-word constituent. With a high-precision tagger that errs on the side of permitting constituents, they show a significant improvement in speed with no reduction in accuracy.

It is difficult to port their idea directly to dependency parsing without committing to a particular search algorithm,[2] and thereby sacrificing some of the graph-based formalism's modularity. However, some of our linear filters (see Section 4.3) were inspired by their constraints.

3.3 Coarse-to-fine Parsing

Another common method employed to speed up exhaustive parsers is a coarse-to-fine approach, where a cheap, coarse model prunes the search space for later, more expensive models (Charniak et al., 2006; Petrov and Klein, 2007). This approach assumes a common forest or chart representation, shared by all granularities, where one can efficiently track the pruning decisions of the coarse models. One could imagine applying such a solution to dependency parsing, but the exact implementation of the coarse pass would vary according to the choice in search algorithm. Our filters are much more modular: they apply to both 1st-order spanning tree parsing and 2nd-order projective parsing, with no modification.

Carreras et al. (2008) use coarse-to-fine pruning with dependency parsing, but in that case, a graph-based dependency parser provides the coarse pass, with the fine pass being a far-more-expensive tree-adjoining grammar. Our filters could become a 0th pass, further increasing the efficiency of their approach.

4 Arc Filters

We propose arc filtering as a preprocessing step for dependency parsing. An arc filter removes implausible head-modifier arcs from the complete dependency graph (which initially includes all head-modifier arcs). We use three stages of filters that operate in sequence on progressively sparser graphs: 1) rule-based, 2) linear: a single pass through the n nodes in a sentence ($O(n)$ complexity), and 3) quadratic: a scoring of all remaining arcs ($O(n^2)$). The less intensive filters are used first, saving time by leaving fewer arcs to be processed by the more intensive systems.

Implementations of our rule-based, linear, and quadratic filters are publicly available at:
`http://code.google.com/p/arcfilter/`

4.1 Filter Framework

Our filters assume the input sentences have been PoS-tagged. We also add an artificial root node to each sentence to be the head of the tree's root. Initially, this node is a potential head for all words in the sentence.

Each filter is a supervised classifier. For example, the quadratic filter directly classifies whether a proposed head-modifier pair is *not* a link in the dependency tree. Training data is created from annotated trees. All possible arcs are extracted for each training sentence, and those that are present in the annotated tree are labeled as class -1, while those not present are $+1$. A similar process generates training examples for the other filters. Since our goal is to only filter very implausible arcs, we bias the classifier to high precision, increasing the cost for misclassifying a true arc during learning.[3]

Class-specific costs are command-line parameters for many learning packages. One can interpret the learning objective as minimizing regularized, weighted loss:

$$\min_{\bar{w}} \frac{1}{2}||\bar{w}||^2 + C_1 \sum_{i:y_i=1} l(\bar{w}, y_i, \bar{x}_i) + C_2 \sum_{i:y_i=-1} l(\bar{w}, y_i, \bar{x}_i) \quad (1)$$

where $l()$ is the learning method's loss function, $\bar{x}_i$ and y_i are the features and label for the ith

[2] Johnson's (2007) split-head CFG could implement this idea directly with little effort.

[3] Learning with a cost model is generally preferable to first optimizing error rate and then thresholding the prediction values to select a high-confidence subset (Joachims, 2005), but the latter approach was used successfully for cell classification in Roark and Hollingshead (2008).

not a h	” “ , . ; | CC PRP$ PRP EX -RRB- -LRB-
no $* \leftarrow m$	EX LS POS PRP$
no $m \rightarrow *$	. RP
not a root	, DT
no $h \leftarrow m$	DT←{DT,JJ,NN,NNP,NNS,.} CD←CD NN←{DT,NNP} NNP←{DT,NN,NNS}
no $m \rightarrow h$	{DT,IN,JJ,NN,NNP}→DT NNP→IN IN→JJ

Table 1: Learned rules for filtering dependency arcs using PoS tags. The rules filter 25% of possible arcs while recovering 99.9% of true links.

training example, $\bar{w}$ is the learned weight vector, and C_1 and C_2 are the class-specific costs. High precision is obtained when $C_2 >> C_1$. For an SVM, $l(\bar{w}, y_i, \bar{x}_i)$ is the standard hinge loss.

We solve the SVM objective using LIBLINEAR (Fan et al., 2008). In our experiments, each filter is a linear SVM with the typical L1 loss and L2 regularization.[4] We search for the best combination of C_1 and C_2 using a grid search on development data. At test time, an arc is filtered if $\bar{w} \cdot \bar{x} > 0$.

4.2 Rule-Based Filtering

Our rule-based filters seek to instantly remove those arcs that are trivially implausible on the basis of their head and modifier PoS tags. We first extract labeled examples from gold-standard trees for whenever a) a word is not a head, b) a word does not have a head on the left (resp. right), and c) a pair of words is not linked. We then trained high-precision SVM classifiers. The only features in $\bar{x}$ are the PoS tag(s) of the head and/or modifier. The learned feature weights identify the tags and tag-pairs to be filtered. For example, if a tag has a positive weight in the not-a-head classifier, all arcs having that node as head are filtered.

The classier selects a small number of high-precision rules, shown in Table 1. Note that the rules tend to use common tags with well-defined roles. By focusing on weighted loss as opposed to arc frequency, the classifier discovers structural zeros (Mohri and Roark, 2006), events which could have been observed, but were not. We consider this an improvement over the frequency-based length thresholds employed previously in tag-specific vine parsing.

4.3 Linear-Time Filtering

In the linear filtering stage, we filter arcs on the basis of single nodes and their contexts, passing through the sentences in linear time. For each node, eight separate classifiers decide whether:

1. It is *not* a head (i.e., it is a leaf of the tree).
2. Its head is on the left/right.
3. Its head is within 5 nodes on the left/right.
4. Its head is immediately on the left/right.
5. It is the root.

For each of these decisions, we again train high-precision SVMs with $C_2 >> C_1$, and filter directly based on the classifier output.

If a word is not a head, all arcs with the given word as head can be pruned. If a word is deemed to have a head within a certain range on the left or right, then all arcs that do not obey this constraint can be pruned. If a root is found, no other words should link to the artificial root node. Furthermore, in a projective dependency tree, no arc will cross the root, i.e., there will be no arcs where a head and a modifier lie on either side of the root. We can therefore also filter arcs that violate this constraint when parsing projectively.

Søgaard and Kuhn (2009) previously proposed a tagger to further constrain a vine parser. Their tags are a subset of our decisions (items 4 and 5 above), and have not yet been tested in a state-of-the-art system.

Development experiments show that if we could perfectly make decisions 1-5 for each word, we could remove 91.7% of the total arcs or 95% of negative arcs, close to the upper bound.

Features

Unlike rule-based filtering, linear filtering uses a rich set of features (Table 2). Each feature is a

[4] We also tried L1-regularized filters. L1 encourages most features to have zero weight, leading to more compact and hence faster models. We found the L1 filters to prune fewer arcs at a given coverage level, providing less speed-up at parsing time. Both L1 and L2 models are available in our publicly available implementation.

PoS-tag features	Other features
tag_i	word_i
tag_i, tag_{i-1}	word_{i+1}
tag_i, tag_{i+1}	word_{i-1}
tag_{i-1}, tag_{i+1}	shape_i
tag_{i-2}, tag_{i-1}	prefix_i
tag_{i+1}, tag_{i+2}	suffix_i
tag_j, Left, $_{j=i-5...i-1}$	i
tag_j, Right, $_{j=i+1...i+5}$	i, n
tag_j, (i-j), $_{j=i-5...i-1}$	n - i
tag_j, (i-j), $_{j=i+1...i+5}$	

Table 2: Linear filter features for a node at position i in a sentence of length n. Each feature is also conjoined (unless redundant) with word_i, tag_i, shape_i, prefix_i, and suffix_i (both 4 letters). The shape is the word normalized using the regular expressions [A-Z]+ $\rightarrow$ A and [a-z]+ $\rightarrow$ a.

Binary features	
sign(h-m)	tags_{hm}
tag_{m-1}, tags_{hm}	tag_{m+1}, tags_{hm}
tag_{h-1}, tags_{hm}	tag_{h+1}, tags_{hm}
sign(h-m), tag_h, word_m	
sign(h-m), word_h, tag_m	
Real features $\Rightarrow$ values	
sign(h-m) $\Rightarrow$ h-m	
tag_h, tag_m $\Rightarrow$ h-m	
tag_k, tags_{hm} $\Rightarrow$ Count($\text{tag}_k \in \text{tags}_{h...m}$)	
word_k, tags_{hm} $\Rightarrow$ Count($\text{word}_k \in \text{words}_{h...m}$)	

Table 3: Quadratic filter features for a head at position h and a modifier at position m in a sentence of length n. Here tags_{hm} = (sign(h-m), tag_h, tag_m), while $\text{tags}_{h...m}$ and $\text{words}_{h...m}$ are all the tags (resp. words) between h and m, but within $\pm$5 positions of h or m.

binary indicator feature. To increase the speed of applying eight classifiers, we use the same feature vector for each of the decisions; learning gives eight different weight vectors, one corresponding to each decision function. Feature extraction is constrained to be $O(1)$ for each node, so that overall feature extraction and classification remain a fast $O(n)$ complexity. Feature extraction would be $O(n^2)$ if, for example, we had a feature for *every* tag on the left or right of a node.

Combining linear decisions

We originally optimized the C_1 and C_2 parameter separately for each linear decision function. However, we found we could substantially improve the collective performance of the linear filters by searching for the optimal combination of the component decisions, testing different levels of precision for each component. We selected a few of the best settings for each decision when optimized separately, and then searched for the best combination of these candidates on development data (testing 12960 combinations in all).

4.4 Quadratic-Time Filtering

In the quadratic filtering stage, a single classifier decides whether each head-modifier pair should be filtered. It is trained and applied as described in Section 4.1.

While theoretically of the same complexity as the parser's arc-scoring function ($O(n^2)$), this process can nevertheless save time by employing a compact feature set. We view quadratic filtering as a light preprocessing step, using only a portion of the resources that might be used in the final scoring function.

Features

Quadratic filtering uses both binary *and* real-valued features (Table 3). Real-valued features promote a smaller feature space. For example, one value can encode distance rather than separate features for different distances. We also generalize the "between-tag features" used in McDonald et al. (2005) to be the count of each tag between the head and modifier. The count may be more informative than tag presence alone, particularly for high-precision filters. We follow Galley and Manning (2009) in using only between-tags within a fixed range of the head or modifier, so that the extraction for each pair is $O(1)$ and the overall feature extraction is $O(n^2)$.

Using only a subset of the between-tags as features has been shown to improve speed but impair parser performance (Galley and Manning, 2009). By filtering quickly first, then scoring all remaining arcs with a cubic scoring function in the parser, we hope to get the best of both worlds.

5 Filter Experiments

Data

We extract dependency structures from the Penn Treebank using the Penn2Malt extraction tool,[5] which implements the head rules of Yamada and Matsumoto (2003). Following convention, we divide the Treebank into train (sections 2–21), development (22) and test sets (23). The development and test sets are re-tagged using the Stanford tagger (Toutanova et al., 2003).

Evaluation Metrics

To measure intrinsic filter quality, we define **Reduction** as the proportion of total arcs removed, and **Coverage** as the proportion of true head-modifier arcs retained. Our evaluation asks, for each filter, what Reduction can be obtained at a given Coverage level? We also give **Time**: how long it takes to apply the filters to the test set (excluding initialization).

We compute an **Upper Bound** for Reduction on development data. There are 1.2 million potential dependency links in those sentences, 96.5% of which are not present in a gold standard dependency tree. Therefore, the maximum achievable Reduction is 96.5%.

Systems

We evaluate the following systems:

- **Rules**: the rule-based filter (Section 4.2)
- **Lin.**: the linear-time filters (Section 4.3)
- **Quad.**: the quadratic filter (Section 4.4)

The latter two approaches run on the output of the previous stage. We compare to the two vine parsing approaches described in Section 3.1:

- **Len-Vine** uses a hard limit on arc length.
- **Tag-Vine** (later, **Vine**) learns a maximum length for dependency arcs for every head/modifier tag-combination and order.

5.1 Results

We set each filter's parameters by selecting a Coverage-Reduction tradeoff on development data (Figure 2). The Lin curve is obtained by varying both the C_1/C_2 cost parameters and the combination of components (plotting the best Reduction at each Coverage level). We chose the linear filters with 99.8% Coverage at a 54.2% Reduction. We apply Quad on this output, varying the cost parameters to produce its curve. Aside from Len-Vine, all filters remove a large number of arcs with little drop in Coverage.

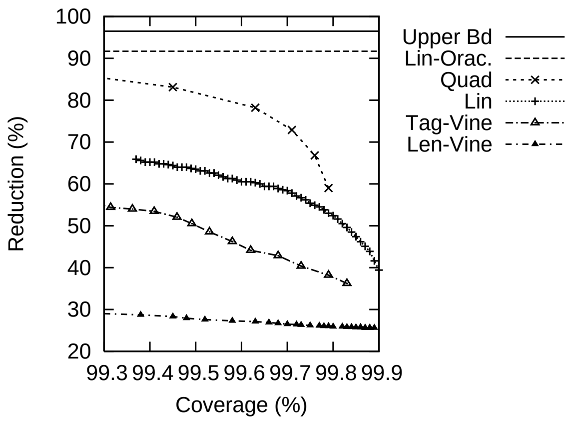


Figure 2: Filtering performance for different filters and cost parameters on development data. Lin-Orac indicates the percentage filtered using perfect decisions by the linear components.

Filter	Coverage	Reduct.	Time (s)
Vine	99.62	44.0	2.9s
Rules	99.86	25.8	1.3s
Lin.	99.73	54.2	7.3s
Quad.	99.50	78.4	16.1s

Table 4: Performance (%) of filters on test data.

After selecting a desired trade-off for each classifier, we move to final filtering experiments on unseen test data (Table 4). The linear filter removes well over half the links but retains an astounding 99.7% of correct arcs. Quad removes 78.4% of arcs at 99.5% Coverage. It thus reduces the number of links to be scored by a dependency parser by a factor of five.

The time for filtering the 2416 test sentences varies from almost instantaneous for Vine and Rules to around 16 seconds for Quad. Speed numbers are highly machine, design, and implemen-

[5] http://w3.msi.vxu.se/~nivre/research/Penn2Malt.html

Decision	Precision	Recall
No-Head	99.9	44.8
Right-∅	99.9	28.7
Left-∅	99.9	39.0
Right-5	99.8	31.5
Left-5	99.9	19.7
Right-1	99.7	6.2
Left-1	99.7	27.3
Root	98.6	25.5

Table 5: Linear Filters: Test-set performance (%) on decisions for components of the combined 54.2 Reduct./99.73 Coverage linear filter.

Type	Coverage	Reduct.	Oracle
All	**99.73**	**54.2**	**91.8**
All\No-Head	99.76	46.4	87.2
All\Left-∅	99.74	53.2	91.4
All\Right-∅	99.75	53.6	90.7
All\Left-5	99.74	53.2	89.7
All\Right-5	99.74	51.6	90.4
All\Left-1	99.75	53.5	90.8
All\Right-1	99.73	53.9	90.6
All\Root	99.76	50.2	90.0

Table 6: Contribution of different linear filters to test set performance (%). Oracle indicates the percentage filtered by perfect decisions.

tation dependent, and thus we have stressed the asymptotic complexity of the filters. However, the timing numbers show that arc filtering can be done quite quickly. Section 6 confirms that these are very reasonable costs in light of the speed-up in overall parsing.

5.2 Linear Filtering Analysis

It is instructive to further analyze the components of the linear filter. Table 5 gives the performance of each classifier on its specific decision. **Precision** is the proportion of positive classifications that are correct. **Recall** is the proportion of positive instances that are classified positively (e.g. the proportion of actual roots that were classified as roots). The decisions correspond to items 1-5 in Section 4.3. For example, *Right-∅* is the decision that a word has *no* head on the right.

Most notably, the optimum *Root* decision has much lower Precision than the others, but this has little effect on its overall accuracy as a filter (Table 6). This is perhaps because the few cases of false positives are still likely to be main verbs or auxiliaries, and thus still still likely to have few links crossing them. Thus many of the filtered links are still correct.

Table 6 provides the performance of the classifier combination when each linear decision is excluded. *No-Head* is the most important component in the oracle and the actual combination.

6 Parsing Experiments

6.1 Set-up

In this section, we investigate the impact of our filters on graph-based dependency parsers. We train each parser unfiltered, and then measure its speed and accuracy once filters have been applied. We use the same training, development and test sets described in Section 5. We evaluate unlabeled dependency parsing using head **accuracy**: the percentage of words (ignoring punctuation) that are assigned the correct head.

The filters bypass feature extraction for each filtered arc, and replace its score with an extremely low negative value. Note that 2$^{\text{nd}}$-order features consider $O(n^3)$ $[h, m_1, m_2]$ triples. These triples are filtered if at least one component arc ($[h, m_1]$ or $[h, m_2]$) is filtered.

In an optimal implementation, we might also have the parser re-use features extracted during filtering when scoring the remaining arcs. We did not do this. Instead, filtering was treated as a preprocessing step, which maximizes the portability of the filters across parsers. We test on two state-of-the art parsers:

MST We modified the publicly-available MST parser (McDonald et al., 2005)[6] to employ our filters before carrying out feature extraction. MST is trained with 5-best MIRA.

DepPercep We also test an in-house dependency parser, which conducts projective first and 2$^{\text{nd}}$-order searches using the split-head CFG described by Johnson (2007), with a weight vector trained using an averaged perceptron (Collins,

[6] `http://sourceforge.net/projects/mstparser/`

Filter	Cost	DepPercep-1 Acc.	DepPercep-1 Time	DepPercep-2 Acc.	DepPercep-2 Time	MST-1 Acc.	MST-1 Time	MST-2 Acc.	MST-2 Time
None	+0	91.8	348	92.5	832	91.2	153	91.9	200
Vine	+3	91.7	192	92.3	407	91.2	99	91.8	139
Rules	+1	91.7	264	92.4	609	91.2	125	91.9	167
Linear	+7	91.7	168	92.4	334	91.2	88	91.8	121
Quad.	+16	91.7	79	92.3	125	91.2	58	91.8	80

Table 7: The effect of filtering on the speed and accuracy on 1^{st} and 2^{nd}-order dependency parsing.

2002). Its features are a mixture of those described by McDonald et al. (2005), and those used in the Koo et al. (2008) baseline system; we do not use word-cluster features.

DepPercep makes some small improvements to MST's 1^{st}-order feature set. We carefully determined which feature types should have distance appended in addition to direction. Also, inspired by the reported utility of mixing PoS tags and word-clusters (Koo et al., 2008), we created versions of all of the "Between" and "Surrounding Word" features described by McDonald et al. (2005) where we mix tags and words.[7]

DepPercep was developed with quadratic filters in place, which enabled a fast development cycle for feature engineering. As a result, it does not implement many of the optimizations in place in MST, and is relatively slow unfiltered.

6.2 Results

The parsing results are shown in Table 7, where times are given in seconds, and **Cost** indicates the additional cost of filtering. Note that the impact of all filters on accuracy is negligible, with a decrease of at most 0.2%. In general, parsing speed-ups mirror the amount of arc reduction measured in our filter analysis (Section 5.1).

Accounting for filter costs, the benefits of quadratic filtering depend on the parser. The extra benefit of quadratic over linear is substantial for DepPercep, but less so for 1^{st}-order MST.

MST shows more modest speed-ups than DepPercep, but MST is already among the fastest publicly-available data-driven parsers. Under quadratic filtering, MST-2 goes from processing 12 sentences per second to 23 sentences.[8]

DepPercep-2 starts slow, but benefits greatly from filtering. This is because, unlike MST-2, it does not optimize feature extraction by factoring its ten 2^{nd}-order features into two triple ($[h, m_1, m_2]$) and eight sibling ($[m_1, m_2]$) features. This suggests that filtering could have a dramatic effect on a parser that uses more than a few triple features, such as Koo et al. (2008).

7 Conclusion

We have presented a series of arc filters that speed up graph-based dependency parsing. By treating filtering as weighted classification, we learn a cascade of increasingly complex filters from tree-annotated data. Linear-time filters prune 54% of total arcs, while quadratic-time filters prune 78%. Both retain at least 99.5% of true dependencies. By testing two state-of-the-art dependency parsers, we have shown that our filters produce substantial speed improvements in even carefully-optimized parsers, with negligible losses in accuracy. In the future we hope to leverage this reduced search space to explore features derived from large corpora.

References

Carreras, Xavier, Michael Collins, and Terry Koo. 2008. TAG, dynamic programming, and the perceptron for efficient, feature-rich parsing. In *CoNLL*.

Carreras, Xavier. 2007. Experiments with a higher-order projective dependency parser. In *EMNLP-CoNLL*.

[7] This was enabled by using word features only when the word is among the 800 most frequent in the training set.

[8] This speed accounts for 25 total seconds to apply the rules, linear, and quadratic filters.

Charniak, Eugene, Mark Johnson, Micha Elsner, Joseph Austerweil, David Ellis, Isaac Haxton, Catherine Hill, R. Shrivaths, Jeremy Moore, Michael Pozar, and Theresa Vu. 2006. Multilevel coarse-to-fine PCFG parsing. In *HLT-NAACL*.

Collins, Michael. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *EMNLP*.

Crammer, Koby and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *JMLR*, 3:951–991.

Dreyer, Markus, David A. Smith, and Noah A. Smith. 2006. Vine parsing and minimum risk reranking for speed and precision. In *CoNLL*.

Eisner, Jason and Noah A. Smith. 2005. Parsing with soft and hard constraints on dependency length. In *IWPT*.

Eisner, Jason. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING*.

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *JMLR*, 9:1871–1874.

Galley, Michel and Christopher D. Manning. 2009. Quadratic-time dependency parsing for machine translation. In *ACL-IJCNLP*.

Joachims, Thorsten. 2005. A support vector method for multivariate performance measures. In *ICML*.

Johnson, Mark. 2007. Transforming projective bilexical dependency grammars into efficiently-parsable CFGs with unfold-fold. In *ACL*.

Koo, Terry, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *ACL-08: HLT*.

McDonald, Ryan and Joakim Nivre. 2007. Characterizing the errors of data-driven dependency parsing models. In *EMNLP-CoNLL*.

McDonald, Ryan and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *EACL*.

McDonald, Ryan, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *ACL*.

Meĺčuk, Igor A. 1987. *Dependency syntax: theory and practice*. State University of New York Press.

Mohri, Mehryar and Brian Roark. 2006. Probabilistic context-free grammar induction based on structural zeros. In *HLT-NAACL*.

Petrov, Slav and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *HLT-NAACL*.

Poon, Hoifung and Pedro Domingos. 2009. Unsupervised semantic parsing. In *EMNLP*.

Roark, Brian and Kristy Hollingshead. 2008. Classifying chart cells for quadratic complexity context-free inference. In *COLING*.

Søgaard, Anders and Jonas Kuhn. 2009. Using a maximum entropy-based tagger to improve a very fast vine parser. In *IWPT*.

Toutanova, Kristina, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL*.

Wang, Qin Iris, Colin Cherry, Dan Lizotte, and Dale Schuurmans. 2006. Improved large margin dependency parsing via local constraints and Laplacian regularization. In *CoNLL*.

Wang, Mengqiu, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for QA. In *EMNLP-CoNLL*.

Wang, Qin Iris, Dale Schuurmans, and Dekang Lin. 2008. Semi-supervised convex training for dependency parsing. In *ACL-08: HLT*.

West, D. 2001. *Introduction to Graph Theory*. Prentice Hall, 2nd edition.

Yamada, Hiroyasu and Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *IWPT*.

A Hierarchical Classifier Applied to Multi-way Sentiment Detection

Adrian Bickerstaffe and Ingrid Zukerman
Faculty of Information Technology
Monash University
bickerstaffe.adrian@gmail.com, Ingrid.Zukerman@monash.edu

Abstract

This paper considers the problem of document-level multi-way sentiment detection, proposing a hierarchical classifier algorithm that accounts for the inter-class similarity of tagged sentiment-bearing texts. This type of classifier also provides a natural mechanism for reducing the feature space of the problem. Our results show that this approach improves on state-of-the-art predictive performance for movie reviews with three-star and four-star ratings, while simultaneously reducing training times and memory requirements.

1 Introduction

A key problem in sentiment detection is to determine the polarity of sentiment in text. Much of the work on this problem has considered binary sentiment polarity (positive or negative) at granularity levels ranging from sentences (Yu and Hatzivassiloglou, 2003; Mao and Lebanon, 2006; McDonald et al., 2007) to documents (Wilson et al., 2005; Allison, 2008).

This paper considers the more general problem of multi-way sentiment classification for discrete, ordinal rating scales, focusing on the document level, i.e., the problem of predicting the "star" rating associated with a review. This is a supervised learning task involving textual reviews that have been tagged with a rating. Ultimately, the goal is to use classifiers which have been trained on tagged datasets to predict the ratings of untagged reviews.

Typical approaches to the rating scale problem include standard k-way classifiers, e.g., (Pang and Lee, 2005). However, these methods do not explicitly account for sample similarities, e.g., the samples with a "four star" rating being more similar to "three star" samples than to "one star" samples. Consequently, these methods generally do not perform well, while methods which incorporate sample similarity information achieve improved performance (Pang and Lee, 2005).

Sample similarity in the multi-way sentiment detection setting has previously been considered by using Support Vector Machines (SVMs) in conjunction with a metric labeling meta-algorithm (Pang and Lee, 2005); by taking a semi-supervised graph-based learning approach (Goldberg and Zhu, 2006); and by using "optimal stacks" of SVMs (Koppel and Schler, 2006). However, each of these methods have shortcomings (Section 2). Additionally, during the learning process, all approaches employ a set of word/punctuation features collected across *all* rating categories. Hence, the number of features may be very large compared to the number of training samples, which can lead to the model overfitting the data.

The main contribution of this paper is the use of hierarchical classifier trees which combine standard binary classifiers to perform multi-way classification (another approach to reduce multi-class classification to binary classifications is described in (Beygelzimer et al., 2009)). The hierarchical classifier accounts for inter-class similarity by

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 62–70,
Beijing, August 2010

means of tree structures which are obtained using inter-class similarity measures in conjunction with a shortest-spanning algorithm. The tree structures reduce training times since they require only $k-1$ nodes for a k-rating problem. Training times are further reduced by the fact that classifier nodes lower in the tree consider fewer rating classes than those higher up, thereby naturally reducing the number of training samples relevant to lower-level nodes. Additionally, the tree structures offer a means to safely cull irrelevant features at non-root nodes of the tree, thus reducing the dimensionality of the training data for these nodes without loss of information. Our experiments show that our new classifier outperforms state-of-the-art methods on average, achieving improvements of up to 7.00% and 7.72% for three-way and four-way classification problems respectively (Section 4).

2 Related Work

Pang and Lee (2005) incorporated information about label similarities using *metric labeling*, where label relations were encoded via a distance metric. The output of standard k-ary classifiers was then modified such that similar items were more likely to be assigned similar labels. Metric labeling required a label-corrected item-similarity function, which was based on the observation that the *Percentage of Positive Sentences* (*PSP*) in reviews increased as their ratings increased. Notice, however, that item similarity was not incorporated into the first stage of classifier training. Metric labeling adjusted the output of the classifiers only after they were trained without considering rating similarities. Our approach accounts for inter-category relationships from the outset of classifier design, rather than addressing this issue with later adjustments.

Goldberg and Zhu (2006) proposed a semi-supervised learning approach to the rating inference problem in scenarios where labeled training data is scarce. Using a graph-based optimisation approach, Goldberg and Zhu demonstrated that the inclusion of unlabeled reviews in the learning process could produce significantly higher prediction accuracy than predictors trained without unlabeled data. This approach outperformed competing methods when it considered relatively small numbers of labeled samples from the four-category movie review dataset (Pang and Lee, 2005). However, the graph-based method did not perform well when a large number of labeled samples was available. Furthermore, Goldberg and Zhu's graph-based learning method was transductive: new samples could not be classified until they were added to the graph — a problem avoided by our approach.

Koppel and Schler (2006) considered neutral examples, which may express a mixed opinion or may not express any opinion at all, in addition to positive/negative samples. Their experiments showed that neutral examples often did not lie close to the positive/negative decision boundary as previously believed. This gave rise to the idea of "optimal stacks" of SVMs, which were pairwise combinations of binary classifiers that distinguish between two categories for the ternary positive/neutral/negative problem (instead of a single binary classifier trained using only positive and negative samples). The search for an optimal stack is exponential in time. Hence, finding suitable stacks is feasible for the ternary problem, but becomes intractable for larger numbers of categories (in the general case).

Snyder and Barzilay (2007) proposed the "Good Grief" algorithm, which considers multiple aspects of a situation (e.g., a restaurant review that covers service, ambiance and food), and yields a prediction that minimises the dissatisfaction (grief) regarding these aspects. This method significantly outperformed baseline methods and individual classifiers. At present, we do not consider separately different aspects of a review — a task we intend to undertake in the future.

3 Multiclass SVM Classifiers

Since SVMs are binary classifiers, they are often employed for binary sentiment detection. However, as seen above, it is not straightforward to use SVMs for multi-way classification, particularly when there is inter-class similarity.

One might initially expect that a hierarchical SVM classifier could be built using pairwise comparisons of adjacent class labels. However, pairwise comparisons alone do not form a complete classifier, raising the question of how to combine pairwise classifications. The standard techniques to build k-way SVM classifiers are OVA and OVO (Hsu and Lin, 2002), and DAGSVM schemes (Platt et al., 2000). An OVA classifier requires k SVMs for a k-category problem, where the i^{th} SVM is trained using all samples from the i^{th} category versus all other samples. A sample is classified by evaluating all k trained SVMs, and the label of the class which maximizes the decision function is chosen. The OVO scheme trains $\frac{k(k-1)}{2}$ classifiers derived from a pairwise comparison of the target categories. A prediction is made by evaluating each SVM and recording "votes" for the favoured category: the class with the most votes is selected as the predicted category. The DAGSVM scheme builds a *Directed Acyclic Graph* (*DAG*) where each non-leaf node has an SVM that discriminates between two classes. A DAGSVM is iteratively constructed in a top-down fashion by forming a list of all the class labels, and creating a decision node that discriminates between the first and last element of the list. This decision node yields two child nodes, each of which omits one of the two classes that were compared. Each of these nodes then discriminates between the first and last element in its list of classes, and so on. This process continues for each decision path until only one element remains in the list. A sample is classified by successively making decisions down the graph until a leaf node is reached. Like OVO, DAGSVM schemes require training $\frac{k(k-1)}{2}$ decision nodes.

All three techniques suffer from long training times — an issue that is exacerbated by large data sets such as our corpus of approximately 5000 movie reviews (Section 4.1). Additional problems associated with these techniques are: (1) there is no bound on the generalisation error of OVA, (2) OVO schemes tend to overfit, and (3) the performance of a DAGSVM relies on the order in which classes are processed. This order is based on the class labels (rather than similarity between samples), and no practical method is known to optimize this order.

Overfitting also arises when the number of features is very large compared to the number of training samples. In this case, the SVM training process may discover a decision plane that separates the training data well, but performs poorly on unseen test samples. While SVM training algorithms use regularisation to address the overfitting problem, research has shown that a careful reduction in feature vector dimensionality can help combat overfitting (Weston et al., 2003).

A fundamental problem with the above three schemes is that the similarity between samples of nearby classes is not considered. Instead, categories are assumed to be independent. This problem may be addressed by considering SVM regression (SVM-R) (Smola and Schölkopf, 1998), where class labels are assumed to come from a discretisation of a continuous function that maps the feature space to a metric space. However, SVM-R, like the SVM schemes described here, trains on the entire feature set for all the classes in the dataset. In the case of sentiment detection, where words and punctuation marks are commonly taken as features, the sheer number of features may overwhelm the number of training samples, and lead to the model overfitting the data. SVM-R also poses the question of how to quantise the regressor's output to produce discrete class predictions.

3.1 The MCST-SVM Classifier

To address the above problems, we build a decision tree of SVMs that reduces the set of possible classes at each decision node, and takes relative class similarity into account during the tree construction process. We construct the decision tree as a Minimum Cost Spanning Tree (MCST), denoted *MCST-SVM*, based on inter-class similarity measured from feature values (Lorena and de Carvalho, 2005). Each of the decision tree leaves corresponds to a target class, and the interior nodes group classes into disjoint sets. For each internal node in the MCST, an SVM is trained to separate all the samples belonging to classes in its left subtree from those in its right subtree. We use linear SVMs, which have been shown to be effective text classifiers (Pang et al., 2002; Pang and Lee, 2005), and set the SVM parameters to match those used in (Pang and Lee, 2005).[1] Figure 1 contrasts

[1]SVMs are implemented using the C/C++ library `liblinear`, a variant of `libsvm` (Chang and Lin, 2001).

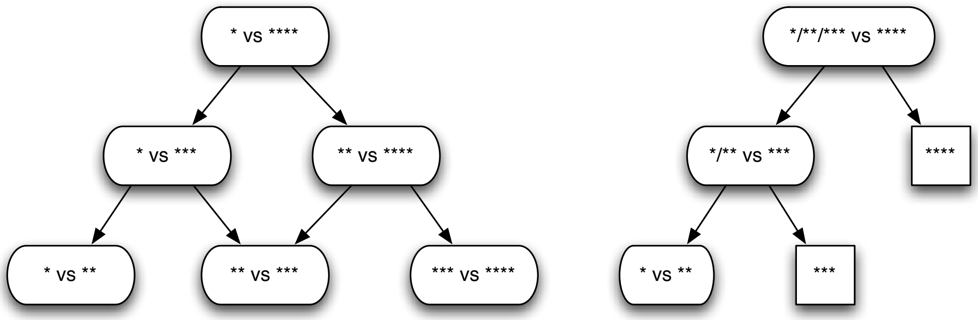


Figure 1: Top section of DAGSVM (left) versus MCST-SVM (right).

the DAGSVM and MCST-SVM approaches for a four-class example.

The MCST is constructed using Kruskal's algorithm (1956), which works in polynomial time (Algorithm 1). This algorithm requires a measure of the similarity between every pair of classes, which is calculated using the distance between a *representative vector* for each class (Section 3.2). The MCST is iteratively built in a bottom-up fashion, beginning with all classes as singleton nodes. In each iteration, the algorithm constructs a node comprising the most similar sets of classes from two previously generated nodes. The similarity between two sets of classes is the shortest distance between the representative vectors of the classes in each set. For instance, the shortest distance between the sets of classes $\{*/**\}$ and $\{***/****\}$ is $\min\{\text{dist}(*,***), \text{dist}(*,****), \text{dist}(**,***), \text{dist}(**,****)\}$. An SVM is then trained to discriminate between the children of the constructed nodes.

With respect to the example in Figure 1, the classes $\{*\}$ and $\{**\}$ are first found to be the most similar, thus forming a node which discriminates between these two classes. In the next iteration, the classes $\{**\}$ and $\{***\}$ are found to be the next most similar, producing a new node which discriminates between $\{*/**\}$ and $\{***\}$. Since the most similar sets are considered lower in the tree, the sets closer to the root of the tree are progressively more dissimilar, until the root node discriminates between the two most dissimilar sets of classes.

Our approach resembles DAGSVMs in that the structure of the decision tree is important. However, unlike DAGSVMs, the MCST-SVM structure is inferred on the basis of similarity between the observed *features* of the data, which are known, rather than the *labels* of the classes, which we are trying to predict. We assume that classes with adjacent labels are similar in the feature space, but if this does not happen in the training data, the MCST-SVM will yield a structure that exploits inter-class similarity irrespective of class labels. Further, our reliance on features supports experimentation with different methods for calculating inter-class similarity (Section 3.2). An additional advantage of MCST-SVM classifiers over the other schemes is that MCST-SVM requires only $k-1$ decision nodes for a k-class problem (and a maximum of $k-1$ decisions to make a prediction). That is, only $k-1$ SVMs must be trained, thereby reducing training time.

3.2 Class Similarity Measures

As mentioned in Section 3.1, the construction of an MCST-SVM classifier requires the computation of a similarity measure between classes. The MCST-SVM method may use any measure of inter-class similarity during the tree construction stage, and many such methods exist (e.g., linear discriminant analysis to order a tree of classifiers (Li et al., 2007)). We elected to use class prototypes to calculate similarity since they have achieved good performance in previous MCST-SVM applications (Lorena and de Carvalho, 2005; Bickerstaffe et al., 2007), and are fast to compute over many documents with a large feature space.

Algorithm 1 Constructing the MCST-SVM

Let V be a set of graph vertices, where each vertex $v_i \in V$ represents rating class i and its available training samples. $\forall i$ compute r_i, the class representative for rating class i.
Let E be a set of graph edges. $\forall i, j$ where $i \neq j$, compute $e_{i,j} \in E$, the distance between class representatives r_i and r_j.
Sort the members of E in ascending order.
$\forall i$, let $S_i = v_i$, and add S_i as a singleton node to the MCST-SVM tree T.
Let $i = 0$ and $j = 0$ be counting variables.
while $i < |V| - 1$ **do**
    Select the j-th edge according to the ordering of inter-class distances.
    if the vertices of the edge are in disjoint sets S_p and S_q **then**
        Define S_p as a positive class and S_q as a negative class.
        Let $S_t = S_p \cup S_q$, and add a new node containing S_t to T.
        Connect the left and right branches of the node containing S_t to the nodes containing S_p and S_q respectively.
        Remove S_p and S_q.
        $i = i + 1$.
    end if
    $j = j + 1$.
end while
Train a binary SVM for each non-leaf node of T.
Return the MCST-SVM tree T.

We first determine a representative feature vector for each class, and then calculate the distance between these representative vectors.

Determining a representative vector. Each review is represented as a vector of boolean attributes, where each attribute indicates the presence or absence of a word or punctuation mark in the text. We elect to use boolean attributes since they have been shown to be advantageous over term-frequency approaches for sentiment detection, particularly when SVMs are employed (Pang et al., 2002). We considered two ways of determining a representative vector: *centroid* and *sample selection*.

- **Centroid.** Given N boolean feature vectors $\boldsymbol{a}_i$ of length n, compute the centroid vector $\boldsymbol{m}$ with values

$$m_j = \frac{1}{N} \sum_{i=1}^{N} a_{i,j} \quad \text{for } j = 1, \ldots, n \; . \qquad (1)$$

  This measure produces a representative vector that contains the proportion of training samples for which each feature occurs.

- **Sample selection.** From the training samples of each class, select one sample which maximises the average *Tanimoto coefficient* (Tanimoto, 1957) with respect to all other samples in that class. The Tanimoto coefficient is an extension of cosine similarity which yields the Jaccard coefficient for boolean feature vectors. Given two boolean vectors $\boldsymbol{a}$ and $\boldsymbol{b}$, the Tanimoto coefficient is defined as

$$d_t(\boldsymbol{a}, \boldsymbol{b}) = \frac{\boldsymbol{a} \cdot \boldsymbol{b}}{\|\boldsymbol{a}\|^2 + \|\boldsymbol{b}\|^2 - \boldsymbol{a} \cdot \boldsymbol{b}} \; , \qquad (2)$$

  where larger values of d_t indicate a higher degree of similarity between boolean vectors. This measure chooses a representative vector which on average has the most "overlap" with all other vectors in the class. We use Tanimoto distance, rather than the classical cosine similarity measure, since we employ boolean valued features instead of term-frequency features.

Calculating distance between vectors. We propose two methods to perform this task: *Euclidean distance* and the *Tanimoto coefficient*.

- **Euclidean distance** is used when the vectors that represent a class are centroid vectors (real-valued).

- The **Tanimoto coefficient** is used when the representative vectors of a class are boolean valued. It is calculated using Equation 2.

3.3 Irrelevant Feature Culling

The MCST-SVM scheme provides a natural mechanism for reducing the dimensionality of feature vectors in order to address the overfitting problem. This is due to the fact that each internal decision node is trained using *only* the samples that belong to the classes relevant to this node. The reviews for these classes are likely to omit some of the words that appear in the reviews for classes that are relevant to other nodes, in particular in the lower layers of the tree. Consequently, an internal node can be trained using a *subset* of the features that occur in the entire training dataset. This subset contains only those features which are observed in the samples relevant to training the node in question.[2] Section 4.2 shows that when tested on "real world" datasets, this method can remove thousands of irrelevant features and improve classifier performance, while reducing memory requirements and training times.

4 Experiments and Results

In this section, we evaluate the MCST-SVM classifier described in Section 3. First, we systematically compare the performance of the different variants of this method: (1) with or without culling irrelevant features, and (2) using the centroid/Euclidean-distance combination or the Tanimoto coefficient to measure inter-class similarity. We then compare the best of these methods with Pang and Lee's (2005). Our results show that a combination of relatively small improvements can achieve a substantial boost in classifier performance, yielding significant improvements over Pang and Lee's results.

All our experiments are performed with 10-fold cross validation, and the results are assessed using classification accuracy.[3] "Significance" refers to statistical significance determined by a paired t-test, with $p < 0.05$.

4.1 Dataset

Our experiments were conducted on the *Sentiment Scale* dataset (v1.0),[4] which comprises four subcorpora of 1770, 902, 1307 and 1027 movie reviews with an associated mapping to a three and four-star rating for each review.[5] Each sub-corpus is written by a different author (denoted Author A, B, C and D respectively), thus avoiding calibration error between individual authors and their ratings. Review texts are automatically filtered to leave only subjective sentences (motivated by the results described in (Pang and Lee, 2004)); the mean number of words per review in each subjective-filtered sub-corpus is 435, 374, 455 and 292 respectively.

4.2 MCST-SVM Variants

Table 1 summarizes the results for the four MCST-SVM variants (the results that are statistically significant compared to the centroid/no-culling option are boldfaced).

Feature culling. Our results show that feature culling produces some improvement in classifier accuracy for all the three-class and four-class datasets. The impact of feature culling is statistically significant for all the four-class datasets when coupled with the Tanimoto coefficient. However, such an effect was not observed for the centroid/Euclidean-distance measure. In the three-class datasets, the improvements from feature culling are marginal for Authors A, B and C, but statistically significant for Author D (4.61%), both when using the centroid/Euclidean-distance measure and the Tanimoto coefficient. We posit that feature culling affects Author D because it reduces the overfitting problem, which caused the initially poor performance of MCST-SVM without culling on this author's short review texts (the reviews by this author, with 292 words on average, are the shortest in the Sentiment Scale dataset by a large margin, Section 4.1). Despite this improvement, all the MCST-SVM variants (as well as Pang and Lee's methods) exhibit worse performance for Authors B and D, who have shorter reviews, than for Authors A and C.

The culling of irrelevant features also has the benefit of reducing node training times and facil-

[2]The root node always considers all classes and therefore considers all features across the whole training dataset.

[3]We also have results for mean absolute error (MAE), which confirm our classification accuracy results.

[4]http://www.cs.cornell.edu/People/pabo/moviereview-data.

[5]In principle, classifiers for the three- and four-class ratings of the Sentiment Scale dataset could be enumerated using optimal stacks of SVMs. However, we wish to directly compare our method with Pang and Lee's (2005). Higher-discrimination datasets (for which optimal stacks are infeasible) will be tested in the future.

	Centroid, no culling	**Tanimoto, no culling**	**Centroid, with culling**	**Tanimoto, with culling**
Three-class				
Author A	70.396	70.396	71.017	71.997
Author B	60.556	60.556	61.111	61.111
Author C	75.154	75.481	76.231	76.923
Author D	59.608	59.608	**64.216**	**64.216**
Four-class				
Author A	62.429	63.810	63.090	**65.720**
Author B	49.111	49.792	50.622	**52.890**
Author C	64.846	65.689	65.692	**66.985**
Author D	49.118	49.626	51.177	**51.873**

Table 1: Performance accuracy (percentage correct predictions) for MCST-SVM variants.

itating a memory-efficient implementation. For example, without feature culling, the nodes of an MCST-SVM for Author A in the four-class dataset take training samples with 19752 features. In contrast, when irrelevant feature culling is applied, the number of features for each of the two non-root decision nodes reduces to 15445 and 17297. This corresponds to a total space saving of 6582 features $((19752 - 15445) + (19752 - 17297))$, yielding an in-memory reduction of 16.7%. Such memory reductions are particularly important for large datasets that may have trouble fitting within typical memory limitations. Node training times are also reduced by up to approximately 10%.

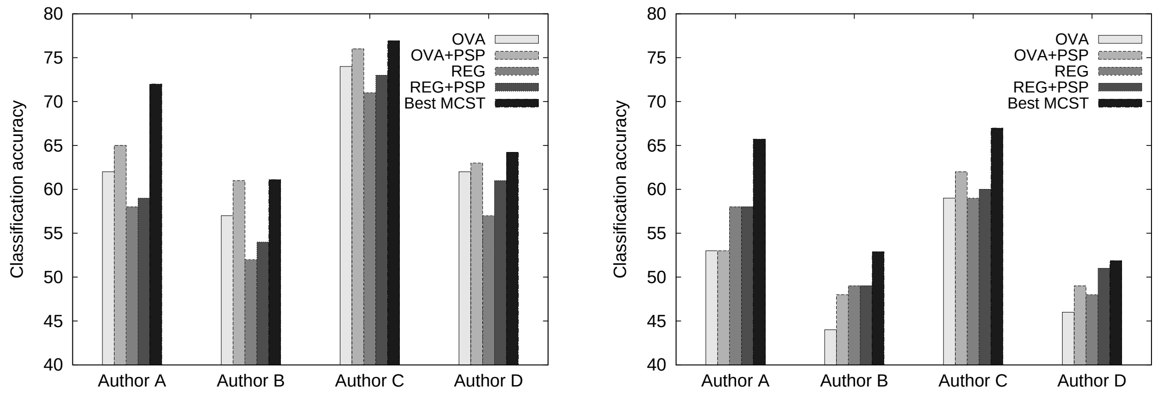
Class similarity measures. As mentioned above, Table 1 shows that the Tanimoto coefficient, coupled with feature culling, yields marginally better results than the centroid/no-culling option for most authors in the three-class dataset, and significantly better results for all the authors in the four-class dataset. The Tanimoto coefficient generally matches or outperforms the centroid/Euclidean-distance measure both with feature culling (Columns 4 and 5 in Table 1) and without feature culling (Columns 2 and 3). However, without feature culling, these improvements are not statistically significant.

For most cases in the three-star dataset, the tree structures found using the Tanimoto coefficient are identical to those found using the Euclidean-centroid option, hence the performance of the classifier is unchanged. For some validation folds, the Tanimoto coefficient discovered tree structures that differed from those found by the Euclidean-centroid option, generally yielding small accuracy improvements (e.g., 0.98% for Author A in the three-star dataset, with feature culling). The Tanimoto coefficient provides a greater benefit for the four-class dataset. Specifically, when feature culling is used (Columns 4 and 5 in Table 1), accuracy improves by 2.63% and 2.27% for Authors A and B respectively (statistically significant), and by 1.29% and 0.70% for Authors C and D respectively. This may be explained by the fact that there are many more tree structures possible for the four-class case than the three-class case, thereby increasing the impact of the inter-class similarity measure for the four-class case. However, this impact is significant only in conjunction with feature culling.

4.3 Comparison with Pang and Lee (2005)

Figure 2 compares the performance of the algorithms presented in (Pang and Lee, 2005) against the performance of the best MCST-SVM variant, which employs feature culling and uses the Tanimoto coefficient to compute inter-class similarity (Section 4.2). As per (Pang and Lee, 2005), REG indicates SVM-R, which is the baseline ordinal regression method. The suffix "+PSP" denotes methods that use the metric labeling scheme. We excluded DAGSVM from our results to maintain consistency with Pang and Lee's experiments. However, according to (Platt et al., 2000), the performance difference between DAGSVM and OVA is not statistically significant.

Generally, the MCST-SVM is competitive against all the classifiers presented in (Pang and Lee, 2005), and in some cases significantly outperforms these methods. Specifically, the hierar-



(a) Three-class data.

(b) Four-class data.

Figure 2: Best MCST-SVM versus competing methods.

chical classifier outperforms OVA+PSP by 7% in the three-class case for Author A (statistically significant), while in the four-class case the MCST-SVM outperforms the best competing methods by 7.72%, 3.89% and 4.98% for Authors A, B, and C respectively (statistically significant). The small improvement of 0.87% for Author D indicates that our approach has the most impact for reviews that contain a relatively large amount of subjective text.

5 Conclusion and Future Work

This paper described a hierarchical classifier applied to multi-way sentiment detection. The classifier is built by exploiting inter-class similarities to arrange high-performance binary discriminators (SVMs) into a tree structure. Since our inter-class similarity measures are based on sample features, they make the problem of structure determination tractable, and enable experimentation with different similarity measures. The resultant structures provide a natural mechanism to remove irrelevant features at each level of the tree, thus reducing the dimensionality of the feature space, which in turn reduces memory requirements. Importantly, these benefits are achieved while improving upon state-of-the-art classification performance, in particular with respect to higher-discrimination datasets.

The MCST-SVM classifier can be generalised to any number of classes, and is extendable in the sense that the classifier algorithm employed in each tree node may be replaced by other classifier algorithms as technology advances. The MCST-SVM classifier is also versatile, and may be applied to variations on the rating classification problem, e.g., traditional text classification.

The MCST-SVM algorithm is not specific to sentiment detection. However, it has several properties which make it particularly suitable for the rating inference problem. Firstly, the MCST-SVM accounts for inter-class similarity and is therefore capable of capturing the ordinal nature of ratings. Secondly, the tree structures permit irrelevant feature culling, which in turn reduces memory requirements and training times.

Future work will involve testing our approach with higher-discrimination datasets, developing methods to pre-process review texts (e.g., improved negation tagging, and incorporating part-of-speech tagging), and further addressing the problem of overfitting. To this effect we will investigate different feature selection algorithms, e.g., (Weston et al., 2003), and their utilisation within the classifier trees. We also propose to consider aspects of reviews (Snyder and Barzilay, 2007), and investigate other methods that measure class similarity, such as selecting typical instances (Zhang, 1992).

Acknowledgments

This research is supported in part by ARC grant LP0883416 and GapBuster Worldwide.

References

Allison, B. 2008. Sentiment detection using lexically-based classifiers. In *Proceedings of the 11th International Conference on Text, Speech and Dialogue*, pages 21–28, Brno, Czech Republic.

Beygelzimer, A., J. Langford, and P. Ravikumar. 2009. Error-correcting tournaments. In *Proceedings of the 20th International Conference on Algorithmic Learning Theory*, pages 247–262, Porto, Portugal.

Bickerstaffe, A., A. Lane, B. Meyer, and K. Marriott. 2007. Building smart diagram environments with domain-specific gesture recognizers. In *Proceedings of the 7th IAPR International Workshop on Graphics Recognition*, pages 145–156, Curitiba, Brazil.

Chang, C.C. and C.J. Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Goldberg, A.B. and X. Zhu. 2006. Seeing stars when there aren't many stars: Graph-based semi-supervised learning for sentiment categorization. In *TextGraphs: Workshop on Graph Based Methods For NLP*, pages 45–52, New York, New York.

Hsu, C. W. and C. J. Lin. 2002. A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425.

Koppel, M. and J. Schler. 2006. The importance of neutral examples for learning sentiment. *Computational Intelligence*, 22(2):100–109.

Kruskal, J. B. 1956. On the shortest spanning subtree and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50.

Li, T., S. Zhu, and M. Ogihara. 2007. Hierarchical document classification using automatically generated hierarchy. *Journal of Intelligent Information Systems*, 29(2):211–230.

Lorena, A. C. and A. C. P. L. F. de Carvalho. 2005. Minimum spanning trees in hierarchical multiclass Support Vector Machines generation. *Innovations in Applied Artificial Intelligence*, 3533:422–431.

Mao, Y. and G. Lebanon. 2006. Isotonic conditional random fields and local sentiment flow. In *Proceedings of the 20th Annual Conference on NIPS*, pages 961–968, British Columbia, Canada.

McDonald, R., K. Hannan, T. Neylon, M. Wells, and J. Reynar. 2007. Structured models for fine-to-coarse sentiment analysis. In *Proceedings of the 45th Annual Meeting of the ACL*, pages 432–439, Prague, Czech Republic.

Pang, B. and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the ACL*, pages 271–278, Barcelona, Spain.

Pang, B. and L. Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the ACL*, pages 115–124, Ann Arbor, Michigan.

Pang, B., L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in NLP*, pages 79–86, Philadelphia, Pennsylvania.

Platt, J. C., N. Cristinini, and J. Shawe-Taylor. 2000. Large margin DAGs for multiclass classification. *Advances in Neural Information Processing Systems*, 12:547–553.

Smola, A. and B. Schölkopf. 1998. A Tutorial on Support Vector regression. Technical Report COLT NC-TR-98-030, University of London.

Snyder, B. and R. Barzilay. 2007. Multiple aspect ranking using the Good Grief algorithm. In *Proceedings of HLT/NAACL*, pages 300–307, Rochester, New York.

Tanimoto, T.T. 1957. IBM internal report.

Weston, J., A. Elisseff, B. Schölkopf, and M. Tipping. 2003. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461.

Wilson, T., J. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the Conference on Empirical Methods in NLP*, pages 347–354, Vancouver, Canada.

Yu, H. and V. Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the Conference on Empirical Methods in NLP*, pages 129–136, Sapporo, Japan.

Zhang, J. 1992. Selecting typical instances in instance-based learning. In *Proceedings of the 9th International Workshop on Machine Learning*, pages 470–479, Aberdeen, Scotland.

Fluency Constraints for Minimum Bayes-Risk Decoding of Statistical Machine Translation Lattices

Graeme Blackwood and **Adrià de Gispert** and **William Byrne**
Machine Intelligence Laboratory, Department of Engineering, Cambridge University
{gwb24|ad465|wjb31}@cam.ac.uk

Abstract

A novel and robust approach to improving statistical machine translation fluency is developed within a minimum Bayes-risk decoding framework. By segmenting translation lattices according to confidence measures over the maximum likelihood translation hypothesis we are able to focus on regions with potential translation errors. Hypothesis space constraints based on monolingual coverage are applied to the low confidence regions to improve overall translation fluency.

1 Introduction and Motivation

Translation quality is often described in terms of *fluency* and *adequacy*. Fluency reflects the 'nativeness' of the translation while adequacy indicates how well a translation captures the meaning of the original text (Ma and Cieri, 2006).

From a purely utilitarian view, adequacy should be more important than fluency. But fluency and adequacy are subjective and not easy to tease apart (Callison-Burch et al., 2009; Vilar et al., 2007). There is a human tendency to rate less fluent translations as less adequate. One explanation is that errors in grammar cause readers to be more critical. A related phenomenon is that the nature of translation errors changes as fluency improves so that any errors in fluent translations must be relatively subtle. It is therefore not enough to focus solely on adequacy. SMT systems must also be fluent if they are to be accepted and trusted. It is possible that the reliance on automatic metrics may have led SMT researchers to pay insufficient attention to fluency: BLEU (Papineni et al., 2002), TER (Snover et al., 2006), and METEOR (Lavie and Denkowski, 2009) show broad correlation with human rankings of MT quality, but are incapable of fine distinctions between fluency and adequacy.

There is concern that the fluency of current SMT is inadequate (Knight, 2007b). SMT is robust, in that a translation is nearly always produced. But unlike translators who should be skilled in at least one of the languages, SMT systems are limited in both source and target language competence. Fluency and accuracy therefore tend to suffer together as translation quality degrades. This should not be the case. Ideally, an SMT system should never be any less fluent than the best *stochastic text generation* system available in the target language (Oberlander and Brew, 2000). What is needed is a good way to enhance the fluency of SMT hypotheses.

The maximum likelihood (ML) formulation (Brown et al., 1990) of translation of source language sentence F to target language sentence $\hat{E}$

$$\hat{E} = \underset{E}{\operatorname{argmax}}\, P(F|E)P(E) \qquad (1)$$

makes it clear why improving SMT fluency is a difficult modelling problem. The language model $P(E)$, the closest thing to a 'fluency component' in the original formulation, only affects candidates likely under the translation model $P(F|E)$. Given the weakness of current translation models this is a severe limitation. It often happens that SMT systems assign $P(F|\bar{E}) = 0$ to a correct reference translation $\bar{E}$ of F (see the discussion in Section 9). The problem is that in ML decoding the language model can only encourage the production of fluent translations; it cannot easily enforce constraints on fluency or introduce new hypotheses.

In Hiero (Chiang, 2007) and syntax-based SMT (Knight and Graehl, 2005; Knight, 2007a), the primary role of syntax is to drive the translation process. Translations produced by these systems respect the syntax of their translation models, but

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 71–79,
Beijing, August 2010

this does not force them to be grammatical in the way that a typical human sentence is grammatical; they produce many translations which are not fluent. The problem is robustness. Generating fluent translations demands a tightly constraining target language grammar but such a grammar is at odds with broad-coverage parsing needed for robust translation.

We have described two problems in translation fluency: (1) SMT may fail to generate fluent hypotheses and there is no simple way to introduce them into the search; (2) SMT produces many translations which are not fluent but enforcing constraints to improve fluency can hurt robustness. Both problems are rooted in the ML decoding framework in which robustness and fluency are conflicting objectives.

We propose a novel framework to improve the fluency of any SMT system, whether syntactic or phrase-based. We will perform Minimum Bayes-risk search (Kumar and Byrne, 2004) over a space of fluent hypotheses $\mathcal{H}$:

$$\hat{E}_{\text{MBR}} = \underset{E' \in \mathcal{H}}{\operatorname{argmin}} \sum_{E \in \mathcal{E}} L(E, E') P(E|F) \qquad (2)$$

In this approach the MBR evidence space $\mathcal{E}$ is generated by an SMT system as a k-best list or lattice. The system runs in its best possible configuration, ensuring both translation robustness and good baselines. Rather than decoding in the output of the baseline SMT system, translations will be sought among a collection of fluent sentences that are close to the top SMT hypotheses as determined by the loss function $L(E, E')$.

Decoupling the MBR hypothesis space from first-pass translation offers great flexibility. Hypotheses in $\mathcal{H}$ may be arbitrarily constrained according to lexical, syntactic, semantic, or other considerations, with no effect on translation robustness. This is because constraints on fluency do not affect the production of the evidence space by the baseline system. Robustness and fluency are no longer conflicting objectives. This framework also allows the MBR hypothesis space to be augmented with hypotheses produced by an NLG system, although this is beyond the scope of the present paper.

This paper focuses on searching out fluent strings amongst the vast number of hypotheses encoded in SMT lattices. Oracle BLEU scores computed over k-best lists (Och et al., 2004) show that many high quality hypotheses are produced by first-pass SMT decoding. We propose reducing the difficulty of enhancing the fluency of complete hypotheses by first identifying regions of high-confidence in the ML translations and using these to guide the fluency refinement process. This has two advantages: (1) we keep portions of the baseline hypotheses that we trust and search for alternatives elsewhere, and (2) the task is made much easier since the fluency of sentence fragments can be refined in context.

In what follows, we use posterior probabilities over SMT lattices to identify useful subsequences in the ML translations (Sections 2 & 3). These subsequences drive the segmentation and transformation of lattices into smaller subproblems (Sections 4 & 5). Subproblems are mined for fluent strings (Section 6), resulting in improved translation fluency (Sections 7 & 8). Our results show that, when guided by the careful selection of subproblems, fluency can be improved with no real degradation of the BLEU score.

2 Lattice MBR Decoding

The formulation of the MBR decoder in Equation (2) separates the hypothesis space from the evidence space. We apply the linearised lattice MBR decision rule (Tromble et al., 2008)

$$\hat{E}_{\text{LMBR}} = \underset{E' \in \mathcal{H}}{\operatorname{argmax}} \left\{ \theta_0 |E'| + \sum_{u \in \mathcal{N}} \theta_u \#_u(E') p(u|\mathcal{E}) \right\}, \qquad (3)$$

where $\mathcal{H}$ is the hypothesis space, $\mathcal{E}$ is the evidence space, $\mathcal{N}$ is the set of all n-grams in $\mathcal{H}$ (typically, $n = 1 \ldots 4$), and θ are constants estimated on held-out data. The quantity $p(u|\mathcal{E})$ is the path posterior probability of n-gram u

$$p(u|\mathcal{E}) = \sum_{E \in \mathcal{E}_u} P(E|F), \qquad (4)$$

where $\mathcal{E}_u = \{E \in \mathcal{E} : \#_u(E) > 0\}$ is the subset of paths containing n-gram u at least once. The path posterior probabilities $p(u|\mathcal{E})$ of Equation (4) can be efficiently calculated (Blackwood et al., 2010) using general purpose WFST operations (Mohri et al., 2002).

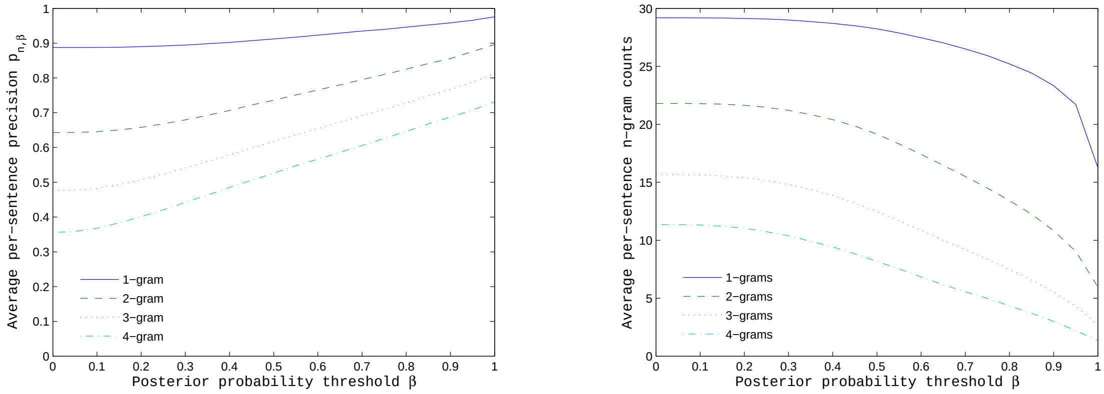


Figure 1: Average n-gram precisions (left) and counts (right) for 2075 sentences of NIST Arabic→English ML translations at a range of posterior probability thresholds $0 \leq \beta \leq 1$. The left plot shows at $\beta = 0$ the n-gram precisions used in the BLEU score of the ML baseline system.

3 Posterior Probability Confidence Measures

In the formulation of Equations (3) and (4) the path posterior n-gram probabilities play a crucial role. MBR decoding under the linear approximation to BLEU is driven mainly by the presence of high posterior n-grams in the lattice; the low posterior n-grams contribute relatively little to the MBR decision criterion. Here we investigate the predictive power of these statistics. We will show that the n-gram posterior is a good predictor as to whether or not an n-gram is to be found in a set of reference translations.

Let $\mathcal{N}_n$ denote the set of n-grams of order n in the ML hypothesis $\hat{E}$, and let $\mathcal{R}_n$ denote the set of n-grams of order n in the union of the references. For confidence threshold β, let $\mathcal{N}_{n,\beta} = \{u \in \mathcal{N}_n : p(u|\mathcal{E}) \geq \beta\}$ denote the n-grams in $\mathcal{N}_n$ with posterior probability greater than or equal to β, where $p(u|\mathcal{E})$ is computed using Equation (4). This is equivalent to identifying all substrings of length n in the translation hypotheses for which the system assigns a posterior probability of β or higher. The precision at order n for threshold β is the proportion of n-grams in $\mathcal{N}_{n,\beta}$ also present in the references:

$$\mathcal{P}_{n,\beta} = \frac{|\mathcal{R}_n \cap \mathcal{N}_{n,\beta}|}{|\mathcal{N}_{n,\beta}|} \tag{5}$$

The left plot in Figure 1 shows average per-sentence n-gram precisions $\mathcal{P}_{n,\beta}$ at orders $1 \ldots 4$ for an Arabic→English translation task at a range of thresholds $0 \leq \beta \leq 1$. Sentence start and end tokens are ignored when computing unigram precisions. We note that precision at all orders improves as the threshold β increases. This confirms that these intrinsic measures of translation confidence have strong predictive power.

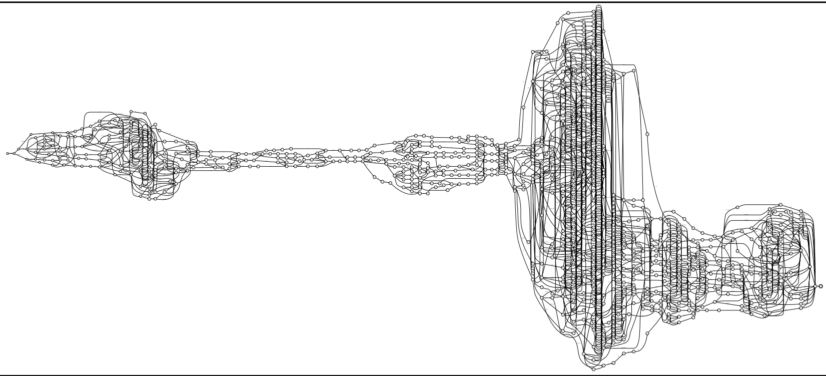
The right-hand side of the figure shows the average number of n-grams per sentence for the same range of β. We see that for high β, there are few n-grams with $p(u|\mathcal{E}) \geq \beta$; this is as expected. However, even at a high threshold of $\beta = 0.9$ there are still on average three 4-grams per sentence with posterior probabilities that exceed β. Even at this very high confidence level, high posterior n-grams occur frequently enough that we can expect them to be useful.

These precision results motivate our use of path posterior n-gram probabilities as a confidence measure. We assign confidence $p(\hat{E}_i^j|\mathcal{E})$ to subsequences $\hat{E}_i \ldots \hat{E}_j$ of the ML hypothesis.

Prior work focuses on word-level confidence extracted from k-best lists and lattices (Ueffing and Ney, 2007), while Zens and Ney (2006) rescore k-best lists with n-gram posterior probabilities. Similar experiments with a slightly different motivation are reported by DeNero et al. (2009); they show that expected n-gram counts in a lattice can be used to predict which n-grams appear in the references.

4 Lattice Segmentation

We have shown that current SMT systems, although flawed, can identify with confidence par-the newspaper “ constitution ” quoted brigadier abdullah krishan **, the chief of police in** karak governorate **(521 km south @-@ west of amman) as saying that the seizure** took place after **police received information that** there were attempts by the group to sell for more than $ 100 thousand dollars , **the police rushed to** the arrest in possession .



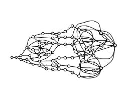
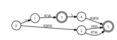
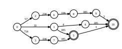
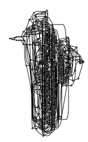
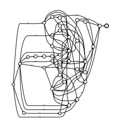
$\mathcal{H}_1$	$\mathcal{H}_2$	$\mathcal{H}_3$	$\mathcal{H}_4$	$\mathcal{H}_5$	$\mathcal{H}_6$	$\mathcal{H}_7$	$\mathcal{H}_8$	$\mathcal{H}_9$
								
433	1	4	1	6	1	6860	1	76

Figure 2: ML translation $\hat{E}$, word lattice $\mathcal{E}$, and decomposition as a sequence of four string and five sublattice regions $\mathcal{H}_1 \ldots \mathcal{H}_9$ using *n*-gram posterior probability threshold $p(u|\mathcal{E}) \geq 0.8$.

tial hypotheses that can be trusted. We wish to constrain MBR decoding to include these trusted partial hypotheses but allow decoding to consider alternatives in regions of low confidence. In this way we aim to improve the best possible output of the best available systems.

We use the path posterior n-gram probabilities of Equation (4) to segment lattice $\mathcal{E}$ into regions of high and low confidence. As shown in the example of Figure 2, the lattice segmentation process is performed relative to the ML hypothesis $\hat{E}$, i.e. relative to the best path through $\mathcal{E}$.

For confidence threshold β, we find all 4-grams $u = \hat{E}_i, \ldots, \hat{E}_{i+3}$ in the ML translation for which $p(u|\mathcal{E}) > \beta$. We then segment $\hat{E}$ into regions of high and low confidence where the high confidence regions are identified by consecutive, overlapping high confidence 4-grams. The high confidence regions are contiguous strings of words for which there is consensus amongst the translations in the lattice. If we trust the path posterior n-gram probabilities, any hypothesised translation should include these high confidence substrings. This approach differs from simple posterior-based pruning in that we discard paths, rather than words or n-grams, which are not consistent with high-confidence regions of the ML hypothesis.

The hypothesis string $\hat{E}$ is in this way segmented into R alternating subsequences of high and low confidence. The segment boundaries are i_r and j_r so that $\hat{E}_{i_r}^{j_r}$ is either a high confidence or a low confidence subsequence. Each subsequence is associated with an unweighted subspace $\mathcal{H}_r$; this subspace has the form of a string for high confidence regions and the form of a lattice for low confidence regions.

If the r^{th} segment is a high confidence region then $\mathcal{H}_r$ accepts only the string $\hat{E}_{i_r}^{j_r}$. If the r^{th} segment is a region of low confidence, then $\mathcal{H}_r$ is built to accept relevant substrings from $\mathcal{E}$. It is constructed as follows. The r^{th} low confidence region $\hat{E}_{i_r}^{j_r}$ has a high confidence left context $\hat{e}_{r-1}$ and a high confidence right context $\hat{e}_{r+1}$ formed from subsequences of the ML translation hypothesis $\hat{E}$ as

$$\hat{e}_{r-1} = \hat{E}_{i_{r-1}}^{j_{r-1}}, \quad \hat{e}_{r+1} = \hat{E}_{i_{r+1}}^{j_{r+1}}$$

Note that when $r = 1$ the left context $\hat{e}_{r-1}$ is the empty string and when $r = R$ the right context $\hat{e}_{r+1}$ is the empty string. We build a transducer

$\mathcal{T}_r$ for the regular expression $/.*\hat{e}_{r-1}(.*)\hat{e}_{r+1}.*/\backslash 1/$.[1] Composition with $\mathcal{E}$ yields $\mathcal{H}_r = \mathcal{E} \circ \mathcal{T}_r$, so that $\mathcal{H}_r$ contains all the reasonable alternatives to $\hat{E}_{i_r}^{j_r}$ in $\mathcal{E}$ consistent with the high confidence left and right contexts $\hat{e}_{r-1}$ and $\hat{e}_{r+1}$. If $\mathcal{H}_r$ is aligned to a high confidence subsequence of $\hat{E}$, we call it a *string region* since it contains a single path; if it is aligned to a low confidence region it is a lattice and we call it a *sublattice region*. The series of high and low confidence subspace regions $\mathcal{H}_1, \ldots, \mathcal{H}_R$ defines the lattice segmentation.

5 Hypothesis Space Construction

We now describe a general framework for improving the fluency of the MBR hypothesis space.

The segmentation of the lattice described in Section 4 considerably simplifies the problem of improving the fluency of its hypotheses since each region of low confidence may be considered independently. The low confidence regions can be transformed one-by-one and then reassembled to form a new MBR hypothesis space.

In order to transform the hypothesis region $\mathcal{H}_r$ it is important to know the context in which it occurs, i.e. the sequences of words that form its prefix and suffix. Some transformations might need only a short context; others may need a sentence-level context, i.e. the full sequence of ML words $\hat{E}_1^{j_r-1}$ and $\hat{E}_{i_{r+1}}^N$ to the left and right of the region $\mathcal{H}_r$ that is to be transformed.

To put this formally, each low confidence sublattice region is transformed by the application of some function Ψ:

$$\mathcal{H}_r \leftarrow \Psi(\hat{E}_1^{j_r-1},\, \mathcal{H}_r,\, \hat{E}_{i_{r+1}}^N) \tag{6}$$

The hypothesis space is then constructed from the concatenation of high confidence string and transformed low confidence sublattice regions

$$\mathcal{H} = \mathcal{E} \circ \bigotimes_{1 \le r \le R} \mathcal{H}_r \tag{7}$$

The composition with the original lattice $\mathcal{E}$ discards any new hypotheses that might be created via the unconstrained concatenation of strings from the $\mathcal{H}_r$. It may be that in some circumstances the introduction of new paths is good, but in what follows we test the ability to improve fluency by searching among existing hypotheses, and this ensures that nothing new is introduced.

Size of the Hypothesis Space If no new hypotheses are introduced by the operations Ψ, the size of the hypothesis space $\mathcal{H}$ is determined by the posterior probability threshold β. Only the ML hypothesis remains at $\beta = 0$, since all its subsequences are of high confidence, i.e. can be covered by n-grams with non-zero path posterior probability. At the other extreme, for $\beta = 1$, it follows that $\mathcal{H} = \mathcal{E}$ and no paths are removed, since any string regions created are formed from subsequences that occur on every path in $\mathcal{E}$.

We can therefore use β to tighten or relax constraints on the LMBR hypothesis space. At $\beta = 0$, LMBR returns only the ML hypothesis; at $\beta = 1$, LMBR is done over the full translation lattice. This is shown in Table 1, where the BLEU score approaches the BLEU score of unconstrained LMBR as β increases.

Note also that the size of the resulting hypothesis space is the product of the number of sequences in the sublattice regions. For Figure 2 at $\beta = 0.8$, this product is $\sim$5.4 billion hypotheses. Even for fairly aggressive constraints on the hypothesis space, many hypotheses remain.

6 Monolingual Coverage Constraints

This section describes one implementation of the transformation function Ψ that we will show leads to improved fluency of machine translation output. This transformation is based on n-gram coverage in a large target language text collection: where possible, we filter the sublattice regions so that they contain only long-span n-grams observed in the text. Our motivation is that large monolingual text collections are good guides to fluency. If a hypothesis is composed entirely of previously seen high order n-grams, it is likely to be fluent and should be favoured.

Initial attempts to identify fluent hypotheses in sublattice regions by ranking according to n-gram LM scores were ineffective. Figure 3 shows the difficulties. We see that both the 4-gram Kneser-Ney and 5-gram stupid-backoff language models

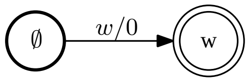
[1] In this notation parentheses indicate string matches so that $/.*y(a*)w.*/\backslash 1/$ applied to $xyaaawzz$ yields aaa.

LM	Translation hypothesis E and n-gram orders used by the LM to score each word	Score
4g	<s>$_1$ the$_2$ reactor$_3$ produces$_3$ plutonium$_2$ *needed*$_2$ *to*$_3$ *manufacture*$_4$ atomic$_3$ bomb$_2$.$_3$ </s>$_4$	-22.59
	<s>$_1$ the$_2$ reactor$_3$ produces$_3$ plutonium$_2$ *needed*$_2$ *to*$_3$ *manufacture*$_4$ *the*$_4$ atomic$_2$ bomb$_3$.$_4$ </s>$_4$	-23.61
5g	<s>$_1$ the$_2$ reactor$_3$ produces$_4$ plutonium$_5$ *needed*$_3$ *to*$_3$ *manufacture*$_4$ atomic$_5$ bomb$_2$.$_3$ </s>$_4$	-16.04
	<s>$_1$ the$_2$ reactor$_3$ produces$_4$ plutonium$_5$ *needed*$_3$ *to*$_3$ *manufacture*$_4$ *the*$_4$ atomic$_4$ bomb$_5$.$_4$ </s>$_5$	-17.96

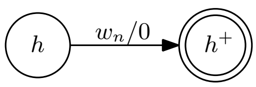
Figure 3: Scores and n-gram orders for hypotheses using 4-gram Kneser-Ney and 5-gram stupid-backoff (estimated from 1.1B and 6.6B tokens, resp.) LMs. Low confidence regions are in italics.

favour the shorter but disfluent hypothesis; normalising by length was not effective. However, the stupid-backoff LM has better coverage and the backing-off behaviour is a clue to the presence of disfluency. Similar cues have been observed in ASR analysis (Chase, 1997). The shorter hypothesis backs off to a bigram for "atomic bomb", whereas the longer hypothesis covers the same words with 4-grams and 5-grams. We therefore disregard the language model scores and focus on n-gram coverage. This is an example where robustness and fluency are at odds. The n-gram models are robust, but often favour less fluent hypotheses.

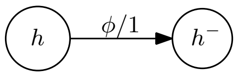
Let $\mathcal{S}$ denote the set of all n-grams in the monolingual training data. To identify partial hypotheses in sublattice regions that have complete monolingual coverage at the maximum order n, we build a coverage acceptor $\mathcal{C}_n$ with a similar form to the WFST representation of an n-gram backoff language model (Allauzen et al., 2003). $\mathcal{C}_n$ assigns a penalty to every n-gram not found in $\mathcal{S}$. In $\mathcal{C}_n$ word arcs have no cost and backoff arcs are assigned a fixed cost of 1. Firstly, arcs from the start state are added for each unigram $w \in \mathcal{N}_1$:



Then for n-grams $u \in \mathcal{S} \cap \{\cup_{i=2}^{n} \mathcal{N}_i\}$, where $u = w_1^n$ consisting of history $h = w_1^{n-1}$ and target word w_n, arcs are added



where $h^+ = w_2^{n-1}$ if u has order n and $h^+ = w_1^n$ if u has order less than n. Backoff arcs are added for each u as



where $h^- = w_2^{n-1}$ if u has order > 2, and bigrams backoff to the null history start state $\emptyset$.

For each sublattice region $\mathcal{H}_r$, we wish to penalise each path proportionally to the number of its n-grams not found in the monolingual text collection $\mathcal{S}$. We wish to do this in context, so that we include the effect of the neighbouring high confidence regions $\mathcal{H}_{r-1}$ and $\mathcal{H}_{r+1}$. Given that we are counting n-grams at order n we form the left context machine $\mathcal{L}_r$ which accepts the *last* $n-1$ words in $\mathcal{H}_{r-1}$; similarly, $\mathcal{R}_r$ accepts the *first* $n-1$ words of $\mathcal{H}_{r+1}$. The concatenation $\mathcal{X}_r = \mathcal{L}_r \otimes \mathcal{H}_r \otimes \mathcal{R}_r$ represents the partial translation hypotheses in $\mathcal{H}_r$ padded with $n-1$ words of left and right context from the neighbouring high confidence regions. Composing $\mathcal{X}_r \circ \mathcal{C}_n$ assigns each partial hypothesis a cost equal to the number of times it was necessary to back off to lower order n-grams while reading the string. Partial hypotheses with cost 0 did not back off at all and contain only maximum order n-grams.

In the following experiments, we look at each $\mathcal{X}_n \circ \mathcal{C}_n$ and if there are paths with cost 0, only these are kept and all others discarded. We introduce this as a constraint on the hypothesis space which we will evaluate for improvement on fluency. Here the transformation function Ψ returns $\mathcal{H}_r$ as $\mathcal{X}_r \circ \mathcal{C}_n$ after pruning. If $\mathcal{X}_r \circ \mathcal{C}_n$ has no zero cost paths, the transformation function Ψ returns $\mathcal{H}_r$ as we find it, since there is not enough monolingual coverage to guide the selection of fluent hypotheses. After applying monolingual coverage constraints to each region, the modified hypothesis space used for MBR search is formed by concatenation using Equation (7).

We note that $\mathcal{C}_n$ is a simplistic NLG system. It generates strings by concatenating n-grams found in $\mathcal{S}$. We do not allow it to run 'open loop' in these experiments, but instead use it to find the strings in $\mathcal{X}_r$ with good n-gram coverage.

7 LMBR Over Segmented Lattices

The effect of fluency constraints on LMBR decoding is evaluated in the context of the NIST Arabic→English MT task. The set *tune* consists

ML	... view , especially with *the open chinese economy* to the world and ...
+LMBR	... view , especially with *the open chinese economy* to the world and ...
+LMBR+CC	... view , especially with *the opening of the chinese economy* to the world and ...
ML	... revision of the constitution *of the japanese public* , which dates back ...
+LMBR	... revision of the constitution *of the japanese public* , which dates back ...
+LMBR+CC	... revision of the constitution *of japan* , which dates back ...

Figure 4: Improved fluency through the application of monolingual coverage constraints to the hypothesis space in MBR decoding of NIST MT 08 Arabic→English newswire lattices.

of the odd numbered sentences of the MT02–MT05 testsets; the even numbered sentences form *test*. MT08 performance on *nw08* (newswire) and *ng08* (newsgroup) data is also reported.

First-pass translation is performed using HiFST (Iglesias et al., 2009), a hierarchical phrase-based decoder. The first-pass LM is a modified Kneser-Ney (Kneser and Ney, 1995) 4-gram estimated over the English side of the parallel text and an 881M word subset of the English GigaWord 3rd Edition. Prior to LMBR, the first-pass lattices are rescored with zero-cutoff stupid-backoff 5-gram language models (Brants et al., 2007) estimated over more than 6B words of English text. The LMBR factors $\theta_0, \ldots, \theta_4$ are set as in Tromble et al. (2008) using unigram precision $p = 0.85$ and recall ratio $r = 0.74$.

The effect of performing LMBR over the segmented hypothesis space is shown in Table 1. The hypothesis subspaces $\mathcal{H}_r$ are constructed at various confidence thresholds as described in Section 4 with $\mathcal{H}$ formed via Equation (7); no coverage constraints are applied yet. Constraining the search space using $\beta = 0.6$ leads to little degradation in LMBR performance under BLEU. This shows lattice segmentation works as intended.

We next investigate the effect of monolingual coverage constraints on BLEU. We build acceptors $\mathcal{C}_n$ as described in Section 6 with $\mathcal{S}$ consisting of all n-grams in the English GigaWord. At $\beta = 0.6$ we found 181 sentences with sublattices $\mathcal{H}_r$ spanned by maximum order n-grams from $\mathcal{S}$, i.e. for which $\mathcal{X}_r \circ \mathcal{C}_n$ have paths with cost 0; these are filtered as described. LMBR over these coverage-constrained sublattices is denoted LMBR+CC. On *nw08* the BLEU score for LMBR+CC is 52.0 which is +0.7 over the ML decoder and only -0.2 BLEU below unconstrained LMBR decoding. Done in this way, constraining hypotheses to have 5-grams from the GigaWord has little impact on BLEU.

		tune	*test*	*nw08*	*ng08*
ML		54.2	53.8	51.3	36.3
β	0.0	54.2	53.8	51.3	36.3
	0.2	54.3	53.8	51.3	36.3
	0.4	54.6	54.2	51.6	36.7
	0.6	54.9	54.4	52.1	36.6
	0.8	54.9	54.4	52.1	36.6
	1.0	54.9	54.4	52.2	36.7
LMBR		54.9	54.4	52.2	36.8

Table 1: BLEU scores for ML hypotheses and LMBR decoding in $\mathcal{H}$ over $0 \leq \beta \leq 1$.

At this value of β, 116 of the 813 *nw08* sentences have a low confidence region (1) completely covered by 5-grams, and (2) within which the ML hypothesis and the LMBR+CC hypothesis differ. It is these regions which we will inspect for improved fluency.

8 Human Fluency Evaluation

We asked 17 native speakers to judge the fluency of sentence fragments from *nw08*. We compared hypotheses from the ML and the LMBR+CC decoders. Each fragment consisted of the partial translation hypothesis from a low confidence region together with its left and right high confidence contexts (examples given in Figure 4). For each sample, judges were asked: “Could this fragment occur in a fluent sentence?”

The results are shown in Table 2. Most of the time, the ML and LMBR+CC sentence fragments were both judged to be fluent; it often happened that they differed by only a single noun or verb substitution which didn’t affect fluency. In a small number of cases, both ML and LMBR+CC were judged to be disfluent. We are most interested in the ‘off-diagonal’ cases. In cases when one system was judged to be fluent and the other was not, LMBR+CC was preferred about twice as often as the ML baseline (26.9% to 9.7%). In other words, the monolingual fluency constraints were judged

		LMBR+CC	
		Fluent	Not Fluent
ML	Fluent	1175 (59.6%)	192 (9.7%)
	Not Fluent	530 (26.9%)	75 (3.8%)

Table 2: Partial hypothesis fluency judgements.

to have improved the fluency of the low confidence region more than twice as often as a fluent hypothesis was made disfluent.

Some examples of improved fluency are shown in Figure 4. Although both the ML and unconstrained LMBR hypotheses might satisfy adequacy, they lack the fluency of the LMBR+CC hypotheses generated using monolingual fluency constraints.

9 Summary and Discussion

We have described a general framework for improving SMT fluency. Decoupling the hypothesis space from the evidence space allows for much greater flexibility in lattice MBR search.

We have shown that high path posterior probability n-grams in the ML translation can be used to guide the segmentation of a lattice into regions of high and low confidence. Segmenting the lattice simplifies the process of refining the hypothesis space since low confidence regions can be refined in the context of their high confidence neighbours. This can be done independently before reassembling the refined regions. Lattice segmentation facilitates the application of post-processing and rescoring techniques targeted to address particular deficiencies in ML decoding.

The techniques we presented are related to consensus decoding and system combination for SMT (Matusov et al., 2006; Sim et al., 2007), and to segmental MBR for automatic speech recognition (Goel et al., 2004). Mohit et al. (2009) describe an alternative approach to improving specific portions of translation hypotheses. They use an SVM classifier to identify a single phrase in each source language sentence that is “difficult to translate”; such phrases are then translated using an adapted language model estimated from parallel data. In contrast to their approach, our approach is able to exploit large collections of monolingual data to refine multiple low confidence regions using posterior probabilities obtained from a high-quality evidence space of first-pass translations.

Testset	Sentences	Reachability
tune	2075	15%
test	2040	14%
nw08	813	11%
ng08	547	9%

Table 3: Arabic→English reference reachability.

We applied hypothesis space constraints based on monolingual coverage to low confidence regions resulting in improved fluency with no real degradation in BLEU score relative to unconstrained LMBR decoding. This approach is limited by the coverage of sublattices using monolingual text. We expect this to improve with larger text collections or in tightly focused scenarios where in-domain text is less diverse.

However, fluency will be best improved by integrating more sophisticated natural language generation. NLG systems capable of generating sentence fragments in context can be incorporated directly into this framework. If the MBR hypothesis space $\mathcal{H}$ contains a generated hypothesis $\bar{E}$ for which $P(F|\bar{E}) = 0$, $\bar{E}$ could still be produced as a translation, since it can be ‘voted for’ by nearby hypotheses produced by the underlying system.

Table 3 shows the proportion of NIST testset sentences that can be aligned to any of the reference translations using our high quality baseline hierarchical decoder with a powerful grammar. The low level of reachability suggests that NLG may be required to achieve high levels of translation quality and fluency. Other rescoring approaches (Kumar et al., 2009; Li et al., 2009) may also benefit from NLG when the baseline is incapable of generating the reference.

We note that our approach could also be used to improve the fluency of ASR, OCR and other language processing tasks where the goal is to produce fluent natural language output.

Acknowledgments

We would like to thank Matt Gibson and the human judges who participated in the evaluation. This work was supported in part under the GALE program of the Defense Advanced Research Projects Agency, Contract No. HR0011-06-C-0022 and the European Union Seventh Framework Programme (FP7-ICT-2009-4) under Grant Agreement No. 247762.

References

Allauzen, Cyril, Mehryar Mohri, and Brian Roark. 2003. Generalized algorithms for constructing statistical language models. In *Proceedings of ACL 2003*.

Blackwood, Graeme, Adrià de Gispert, and William Byrne. 2010. Efficient path counting transducers for minimum Bayes-risk decoding of statistical machine translation lattices. In *Proceedings of ACL 2010*.

Brants, Thorsten, Ashok C. Popat, Peng Xu, Franz J. Och, and Jeffrey Dean. 2007. Large language models in machine translation. In *Proceedings of the EMNLP 2007*.

Brown, Peter F., John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

Callison-Burch, Chris, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *WMT 2009*.

Chase, Lin Lawrance. 1997. Error-responsive feedback mechanisms for speech recognizers, Ph.D. Thesis, Carnegie Mellon University.

Chiang, David. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

DeNero, John, David Chiang, and Kevin Knight. 2009. Fast consensus decoding over translation forests. In *Proceedings of ACL-IJCNLP 2009*.

Goel, V., S. Kumar, and W. Byrne. 2004. Segmental minimum Bayes-risk decoding for automatic speech recognition. *IEEE Transactions on Speech and Audio Processing*, 12:234–249.

Iglesias, Gonzalo, Adrià de Gispert, Eduardo R. Banga, and William Byrne. 2009. Hierarchical phrase-based translation with weighted finite state transducers. In *Proceedings of the 2009 Annual Conference of the NAACL*.

Kneser, R. and H. Ney. 1995. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing*.

Knight, K and J Graehl. 2005. An overview of probabilistic tree transducers for natural language processing. In *Proceedings of CICLING 2005*.

Knight, K. 2007a. Capturing practical natural language transformations. *Machine Translation*, 21(2).

Knight, Kevin. 2007b. Automatic language translation generation help needs badly. In *MT Summit XI Workshop on Using Corpora for NLG: Keynote Address*.

Kumar, Shankar and William Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *NAACL 2004*.

Kumar, Shankar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. Efficient minimum error rate training and minimum bayes-risk decoding for translation hypergraphs and lattices. In *Proceedings of ACL-IJCNLP 2009*.

Lavie, Alon and Michael J. Denkowski. 2009. The METEOR metric for automatic evaluation of machine translation. *Machine Translation Journal*.

Li, Zhifei, Jason Eisner, and Sanjeev Khudanpur. 2009. Variational decoding for statistical machine translation. In *Proceedings of ACL-IJCNLP 2009*.

Ma, Xiaoyi and Christopher Cieri. 2006. Corpus support for machine translation at LDC. In *LREC 2006*.

Matusov, Evgeny, Nicola Ueffing, and Hermann Ney. 2006. Computing consensus translation from multiple machine translation systems using enhanced hypotheses alignment. In *11th Conference of the EACL*.

Mohit, B., F. Liberato, and R. Hwa. 2009. Language model adaptation for difficult-to-translate phrases. In *Proceedings of the 13th Annual Conference of the EAMT*.

Mohri, Mehryar, Fernando Pereira, and Michael Riley. 2002. Weighted finite-state transducers in speech recognition. In *CSL*, volume 16, pages 69–88.

Oberlander, Jon and Chris Brew. 2000. Stochastic text generation. In *Philosophical Transactions of the Royal Society*.

Och, F., D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. 2004. A smorgasbord of features for statistical machine translation. In *Proceedings of the HLT Conference of the NAACL*.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL 2002*.

Sim, K.-C., W. Byrne, M. Gales, H. Sahbi, and P.C. Woodland. 2007. Consensus network decoding for statistical machine translation system combination. In *ICASSP 2007*.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, , and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of AMTA*.

Tromble, Roy, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice minimum Bayes-risk decoding for statistical machine translation. In *Proceedings of the 2008 Conference on EMNLP*.

Ueffing, Nicola and Hermann Ney. 2007. Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40.

Vilar, D, G Leusch, H Ney, and R Banchs. 2007. Human evaluation of machine translation through binary system comparisons. In *Proceedings of WMT 2007*.

Zens, Richard and Hermann Ney. 2006. N -gram posterior probabilities for statistical machine translation. In *Proceedings of WMT 2006*.

Self-Annotation for Fine-Grained Geospatial Relation Extraction

Andre Blessing **Hinrich Schütze**
Institute for Natural Language Processing
Universität Stuttgart
ner@ifnlp.org

Abstract

A great deal of information on the Web is represented in both textual and structured form. The structured form is machine-readable and can be used to augment the textual data. We call this augmentation – the annotation of texts with relations that are included in the structured data – *self-annotation*. In this paper, we introduce self-annotation as a new supervised learning approach for developing and implementing a system that extracts fine-grained relations between entities. The main benefit of self-annotation is that it does not require manual labeling. The input of the learned model is a representation of the free text, its output structured relations. Thus, the model, once learned, can be applied to any arbitrary free text. We describe the challenges for the self-annotation process and give results for a sample relation extraction system. To deal with the challenge of fine-grained relations, we implement and evaluate both shallow and deep linguistic analysis, focusing on German.

1 Introduction

In the last years, information extraction has become more important in domains like context-aware systems (e.g. Nexus (Dürr et al., 2004)) that need a rich knowledge base to make the right decisions in different user contexts. Geospatial data are one of the key features in such systems and need to be represented on different levels of detail. Data providers do not cover all these levels completely. To overcome this problem, *fine-grained* information extraction (IE) methods can be used to acquire the missing knowledge. We define fine-grained IE as methods that recognize entities at a finer grain than standard categories like person, location, and organization. Furthermore, the quality of the data in context-aware systems plays an important role and updates by an information extraction component can increase the overall user acceptance.

For both issues an information extraction system is required that can handle *fine-grained relations*, e.g., "X is a suburb of Y" or "the river X is a tributary of Y" – as opposed to simple containment. The World Wide Web offers a wealth of information about geospatial data and can be used as source for the extraction task. The extraction component can be seen as a kind of sensor that we call *text senor* (Blessing et al., 2006).

In this paper, we address the problem of developing a flexible system for the acquisition of relations between entities that meets the above desiderata. We concentrate on *geospatial* entities on a fine-grained level although the approach is in principle applicable to any domain. We use a supervised machine learning approach, including several features on different linguistic levels, to build our system. Such a system highly depends on the quality and amount of labeled data in the training phase. The main contribution of this paper is the introduction of self-annotation, a novel approach that allows us to eliminate manual labeling (although training set creation also involves costs other than labeling). Self-annotation is based on the fact that Word Wide Web sites like Wikipedia include, in addi-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 80–88,
Beijing, August 2010

tion to unstructured text, structured data. We use structured data sources to automatically annotate unstructured texts. In this paper, we use German Wikipedia data because it is a good source for the information required for our context-aware system and show that a system created without manual labeling has good performance.

Our trained model only uses text, not the structured data (or any other markup) of the input documents. This means that we can train an information extractor on Wikipedia and then apply it to any text, regardless of whether this text also contains structured information.

In the first part of this paper, we discuss the challenges of self-annotation including some heuristics which can easily be adapted to different relation types. We then describe the architecture of the extraction system. The components we develop are based on the UIMA (Unstructured Information Management Architecture) framework (Hahn et al., 2008) and include two linguistic engines (OpenNLP[1], FSPar). The extraction task is performed by a supervised classifier; this classifier is also implemented as a UIMA component and uses the ClearTK framework. We evaluate our approach on two types of fine-grained relations.

2 Related work

Jiang (2009) also addresses the issue of supervised relation extraction when no large manually labeled data set is available. They use only a few seed instances of the target relation type to train a supervised relation extraction system. However, they use multi-task transfer learning including a large amount of labeled instances of other relation types for training their system. In contrast, our work eliminates manual labeling by using structured data to annotate the relations.

Wu and Weld (2007) extract facts from infoboxes and link them with their corresponding representation in the text. They discuss several issues that occur when using infoboxes as a knowledge base, in particular, (i) the fact that infoboxes are incomplete; and (ii) *schema drift*. Schema drift occurs when authors over time use different attribute names to model facts or the same attributes are used to model different facts. So the semantics of the infoboxes changes slightly and introduces noise into the structured information. Their work differs from self-annotation in that they are not interested in the creation of self-annotated corpora that can be used as training data for other tasks. Their goal is to develop methods that make infoboxes more consistent.

Zhang and Iria (2009) use a novel entity extraction method to automatically generate gazetteers from seed lists using Wikipedia as knowledge source. In contrast to our work they need structured data for the extraction while our system focuses on the extraction of information from unstructured text. Methods that are applicable to any unstructured text (not just the text in the Wikipedia) are needed to increase coverage beyond the limited number of instances covered in Wikipedia.

Nothman et al. (2009) also annotate Wikipedia's unstructured text using structured data. The type of structured data they use is hyperlinking (as opposed to infoboxes) and they use it to derive a labeled named entity corpus. They show that the quality of the annotation is comparable to other manually labeled named entity recognition gold standards. We interpret their results as evidence that self-annotation can be used to create high quality gold standards.

3 Task definition

In this section, we describe the annotation task; give a definition of the relation types covered in this paper; and introduce the extraction model.

We focus on binary relations between two relation arguments occurring in the same sentence. To simplify the self-annotation process we restrict the first argument of the relation to the main entity of the Wikipedia article. As we are building text sensors for a context aware system, relations between geospatial entities are of interest. Thus we consider only relations that use a geospatial named entity as second argument.

We create the training set by automatically identifying all correct binary relations in the text. To this end, we extract the relations from the structured part of the Wikipedia, the infoboxes. Then we automatically find the corresponding

[1] http://opennlp.sourceforge.net/

sentences in the text and annotate the relations (see section 4). All other not yet marked binary relations between the main entity and geospatial entities are annotated as negative samples. The result of this step is a self-annotated training set.

In the second step of our task, the self-annotated training set is used to train the extraction model. The model only takes textual features as input and can be applied to any free text.

3.1 Classification task and relations used

Our relation extraction task is modeled as a classification task which considers a pair of named entities and decides whether they occur in the requested relation or not. The classifier uses extracted features for this decision. Features belong to three different classes. The first class contains *token-based features* and their linguistic labels like part-of-speech, lemma, stem. In the second class, we have *chunks* that aggregate one or more tokens into complex units. *Dependency relations* between the tokens are represented in the third class.

Our classifier is applicable to a wide spectrum of geospatial relation types. For the purposes of a focused evaluation, we selected two relations. The first type contains rivers and the bodies of water into which they flow. We call it *river-bodyOfWater* relation. Our second type is composed of relations between towns and the corresponding suburb. We call this *town-suburb* relation.

3.2 Wikipedia as resource

Wikipedia satisfies all corpus requirements for our task. It contains a lot of knowledge about geospatial data with unstructured (textual) and structured information. We consider only German Wikipedia articles because our target application is a German context aware system. In relation extraction for German, we arguably face more challenges – e.g., more complex morphology and freer word order – than we would in English.

For this work we consider only a subset of the German Wikipedia. We use all articles that belong to the following categories: Rivers by country, Mountains by country, Valleys by country, Islands by country, Mountain passes by country, Forests by country and Settlements by country.

For the annotation task we use the structural content of Wikipedia articles. Most articles belonging to the same categories use similar templates to represent structured information. One type of template is the infobox, which contains pairs of attributes and their values. These attribute-value pairs specify a wide range of geospatial relation types including fine-grained relations. In this work we consider only the infobox data and the article names from the structured data.

For context-aware systems fine-grained relation types are particularly relevant. Such relations are not represented in resources like DBPedia (Auer et al., 2007) or Yago (Suchanek et al., 2007) although they also consist of infobox data. Hence, we have to build our own extraction component (see section 5.2) when using infoboxes.

4 Self-Annotation

Self-annotation is a two-fold task. First, the structured data, in our case the infoboxes of Wikipedia articles, must be analyzed to get all relevant attribute-value pairs. Then all relevant geospatial entities are marked and extracted. In a second step these entities must be matched with the unstructured data.

In most cases, the extraction of the named entities that correspond to the required relations is trivial because the values in the infoboxes consist only of one single entity or one single link. But in some cases the values contain mixed content which can include links, entities and even free text. In order to find an accurate extraction method for those values we have developed several heuristics. See section 5.2 for discussion.

The second task links the extracted structured data to tokens in the textual data. Pattern based string matching methods are not sufficient to identify all relations in the text. In many cases, morphological rules need to be applied to identify the entities in the text. In other cases, the preprocessed text must be retokenized because the borders of multi-word expressions are not consistent with the extracted names in step one. One other issue is that some named entities are a subset of other named entities (*Lonau* vs. *kleine Lonau*;

Gollach	
Die Gollach bei Aub-Baldersheim	
Daten	
Lage	Bayern (Mittelfranken, Unterfranken), Deutschland
Gewässerkennzahl	DE: 2462
Länge	29,06 km
Quelle	zwischen den *Markt Nordheimer* Ortsteilen *Herbolzheim* und *Ulsenheim* am südwestlichen Rand des Steigerwaldes 49° 33′ 34″ N, 10° 18′ 43″ O
Quellhöhe	337,6 m
Mündung	bei Bieberehren (am Ende des Gollachtals) in die Tauber 49° 31′ 14″ N, 10° 0′ 2″ O

Figure 1: Infobox of the German Wikipedia article about *Gollach*.

Gollach

Die **Gollach** ist ein rechter Nebenfluss der **Tauber** in Mittel- und Unterfranken.

Die **Gollach** ist etwa 29 km lang und entsteht zwischen Herbolzheim und Ulsenheim am südwestlichen Rand des Steigerwaldes auf 337,6 m. **Sie** fließt in westlicher Richtung an Ulsenheim (Markt Nordheim), der Kleinstadt Uffenheim und den Orten Gollachostheim (Gollhofen), Lipprichhausen (Hemmersheim) und Hemmersheim vorbei zur Kleinstadt Aub. Nach Aub zieht **sie** dann in südwestliche Richtung und schneidet sich dabei tief in das nach ihr benannte Gollachtal ein. Schließlich mündet **sie** in **Bieberehren** auf 244 m in die **Tauber**.

Die Landschaft um die **Gollach** wird Gollachgau genannt; nach ihr heißen auch die Orte Gollhofen und Gollachostheim. Das Einzugsgebiet umfasst ca. 160 Quadratkilometer, nach Norden und Osten wird es von denen einiger Nebenflüsse des Mains begrenzt, insbesondere der Aisch, im Süden und Westen konkurrieren andere Nebenflüsse der **Tauber** mit ihr.

Figure 2: Textual content of the German Wikipedia article about *Gollach*. All named entities which are relevant for the river-bodyOfWater relation are highlighted. This article contains two instances for the relation between *Gollach* and *Tauber*.

similar to *York* vs. *New York*). We have to use a longest match strategy to avoid such overlapping annotations.

The main goal of the self-annotation task is to reach the highest possible annotation quality. Thus, only complete extracted relations are used for the annotation process while incomplete data are excluded from the training set. This procedure reduces the noise in the labeled data.

4.1 Example

We use the river-bodyOfWater relation between the two rivers *Gollach* and *Tauber* to describe the self-annotation steps.

Figure 1 depicts a part of the infobox for the German Wikipedia article about the river *Gollach*. For this relation the attribute Mündung 'mouth' is relevant. The value contains unstructured information (i.e., text, e.g. *bei 'at' Bieberehren*) and structured information (the link from *Bieberehren* to its Wikipedia page). The relation we want to extract is that the river *Gollach* flows into the river *Tauber*.

Figure 2 shows the textual content of the *Gollach* article. We have highlighted all relevant named entities for the self-annotation process. This includes the name of the article and instances of the pronoun *sie* referring to *Gollach*. Our matching algorithm identifies two sentences as positive samples for the relation between *Gollach* and *Tauber*:

- (i) Die *Gollach* ist ein rechter Nebenfluss der *Tauber* in Mittel- und Unterfranken. (The *Gollach* is a right tributary of the *Tauber* in Middle and Lower Franconia.)
- (ii) Schließlich mündet *sie* in Bieberehren auf 244 m in die *Tauber*. (Finally, *it* discharges in Bieberehren at 244 m above MSL into the *Tauber*.)

5 Processing

In this section we describe how the self-annotation method and relation extraction is implemented. First we introduce the interaction with the Wikipedia resource to acquire the structured and unstructured information for the processing pipeline. Second we present the components of the UIMA pipeline which are used for the relation extraction task.

5.1 Wikipedia interaction

We use the JWPL API (Zesch et al., 2008) to pre-process the Wikipedia data. This interface provides functions to extract structured and unstructured information from Wikipedia. However, many Wikipedia articles do not adhere to valid Wikipedia syntax (missing closing brackets etc.). The API also does not correctly handle all Wikipedia syntax constructions. We therefore have enhanced the API for our extraction task to get high quality data for German Wikipedia articles.

5.2 Infobox extraction

As discussed in section 4 infoboxes are the key resource for the self-annotation step. However the processing of infoboxes that include attribute-value pairs with mixed content is not trivial.

For each new relation type an initial manual effort is required. However, in comparison to the complete annotation of a training corpus, this effort is small. First the attributes used in the infoboxes of the Wikipedia articles relevant for a specific relation have to be analyzed. The results of this analysis simplify the choice of the correct attributes. Next, the used values of these attributes must be investigated. If they contain only single entries (links or named entities) the extraction is trivial. However, if they consist of mixed content (see section 4.1) then specific extraction methods have to be applied. We investigated different heuristics for the self-annotation process to get a method that can easily be adapted to new relation types.

Our first heuristic includes a set of rules specifying the extraction of the values from the infoboxes. This heuristic gives an insufficient basis for the self-annotation task because the rich morphology and free word order in German can not be modeled with simple rules. Moreover, handcrafted rules are arguably not as robust and maintainable as a statistical classifier trained on self-annotated training material.

Our second heuristic is a three step process. In step one we collect all links in the mixed content and replace them by a placeholder. In the second step we tag the remaining content with the OpenNLP tokenizer to get all named entities. Both collected lists are then looked up in a lexicon that contains named entities and the corresponding geospatial classes. This process requires a normalization procedure that includes the application of morphological methods. The second method can be easily adapted to new relation types.

5.3 UIMA

The self-annotated corpora are processed by several components of the UIMA (Müller et al., 2008) pipeline. The advantage of exchangeable collection readers is that they seamlessly handle structured and unstructured data. Another advantage of using UIMA is the possibility to share components with other research groups. We can easily exchange different components, like the usage of the commonly known OpenNLP processing tools or the FSPar NLP engine (Schiehlen, 2003) (which includes the TreeTagger (Schmid, 1995)). This allows us to experiment with different approaches, e.g., shallow vs. deep analysis. The components we use provide linguistic analysis on different levels: tokens, morphology, part of speech (POS), chunking and partial dependency analysis. Figure 4 shows the results after the linguistic processing of our sample sentence. For this work only a few annotations are wrapped as UIMA types: token (incl. lemma, POS), multiword, sentence, NP, PP and dependency relations (labeled edges between tokens). We will introduce our machine learning component in section 5.5. Finally, the CAS consumers allow us to store extracted facts in a context model.

Figure 3 shows the article about *Gollach* after linguistic processing. In the legend all annotated categories are listed. We highlighted all marked relations, all references to the article name (referred to as subject in the figure) and links. After selection of the *Tauber* relation, all annotations for this token are listed in the right panel.

5.4 Coreference resolution

Using anaphora to refer to the main entity is a common practice of the authors of Wikipedia ar-

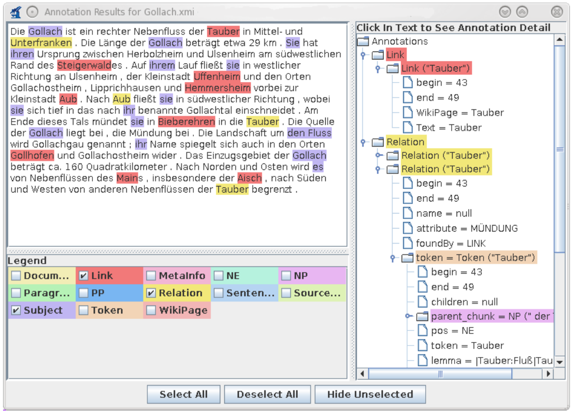


Figure 3: Screenshot of the UIMA Annotation-Viewer.

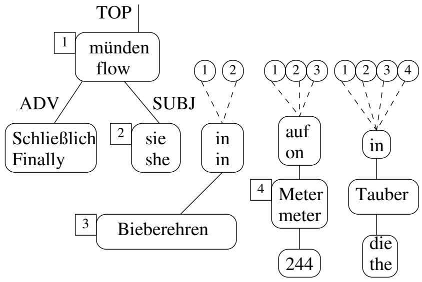


Figure 4: Dependency parser output of the FSPar framework.

	linguistic effort	description
F1	pos-tagging	window size 3, LEMMA
F2	chunk-parse	parent chunks
F3	dependency-parse	dependency paths betw. NEs

Table 1: List of feature types

ticles. Coreference resolution is therefore necessary for our annotation task. A shallow linguistic analysis showed that the writing style is similar throughout Wikipedia articles. Based on this observation, we empirically investigated some geospatial articles and came to the conclusion that a simple heuristic is sufficient for our coreference resolution problem. In almost all articles, pronouns refer to the main entity of the article. In addition we include some additional rules to be able to establish coreference of markables such as *der Fluss* 'the river' or *der Bach* 'the creek' with the main entity.

5.5 Supervised relation extraction

We use the ClearTK (Ogren et al., 2008) toolkit, which is also an UIMA component, for the relation extraction task. It contains wrappers for different machine learning suites. Our initial experiments showed that the MaximumEntropy classifier achieved the best results for our classification task. The toolkit provides additional extensible feature methods. Because we view self-annotation and fine-grained named entity recognition as our main contributions, not feature selection, we only give a brief overview of the features we use.

F1 is a window based bag-of-words feature (window size = 3). It considers lemma and part-of-speech tag of the tokens. F2 is a phrase based extractor that uses the parent phrase of both entities (max 2 levels). F3 is a representation of all possible dependency paths between the article's main entity and a target entity, where each path is represented as a feature vector. In most cases, more than one path is returned by the partial dependency parser (which makes no disambiguation decisions) and included in the feature representation. Figure 4 depicts the dependency parser output of our sample sentence. Each pair of square and circle with the same number corresponds to one dependency. These different possible dependency combinations give rise to 8 possible paths between the relation entities *Tauber* and *sie 'she'* although our example sentence is a very simple sentence.

6 Evaluation

We evaluate the system in two experiments. The first considers the relation between suburbs and their parent towns. In the second experiment the river-bodyOfWater relation is extracted. The experiments are based on the previously described extracted Wikipedia corpus. For each experiment a new self-annotated corpus is created that is split into three parts. The first part (60%) is used as training corpus. The second part (20%) is used as development corpus. The remaining 20% is used for the final evaluation and was not inspected while we were developing the extraction algorithms.

6.1 Metric used

Our gold standard includes all relations of each article. Our metric works on the level of type and is independent of how often the same relation occurs in the article. The metric counts a relation as true positive (TP) if the system extracted it at least once. If the relation was not found by the system a false negative (FN) is counted. A false positive (FP) is given if the system extracts a relation between two entities that is not part of the (infobox-derived) gold standard for the article. All three measures are used to calculate precision ($P = \frac{TP}{TP+FP}$), recall ($R = \frac{TP}{TP+FN}$), and F_1-score ($F_1 = 2\frac{P*R}{P+R}$).

6.2 Town-suburb extraction

The town-suburb extractor uses one attribute of the infobox to identify the town-suburb relation. There is no schema drift in the infobox data and the values contain only links. Therefore the self-annotation works almost perfectly. The only exceptions are articles without an infobox which cannot be used for training. However, this is not a real issue because the amount of remaining data is sufficient: 9000 articles can be used for this task. The results in table 2 show that the classifier that uses F1, F2 and F3 (that is, including the dependency features) performs best.

engine	features	F_1	recall	precision
FSPar	F1	64.9	79.0%	55.7%
FSPar	F1, F2	89.6	90.2%	89.5%
FSPar	F1, F2, F3	98.3	98.8%	97.8%

Table 2: Results of different feature combinations on the test set for town-suburb relation

6.3 River-bodyOfWater extraction

For the extraction of the river-bodyOfWater relation the infobox processing is more difficult. We have to handle more attributes because there is schema drift between the different users. It is hence necessary to merge information coming from different attribute values. The other difficulty is the usage of mixed contents in the values. Another main difference to the town-suburb relation is that the river-bodyOfWater relation is often not mentioned in the first sentence (which usually gives a short definition about the the main entity). Thus, the self-annotation method has to deal with the more complex sentences that are common later in the article. This also contributes to a more challenging extraction task.

Our river-bodyOfWater relation corpus consists of 3000 self-annotated articles.

Table 3 shows the performance of the extractor using two different linguistic components as described in section 5.3. As in the case of town-suburb extraction the classifier that uses all features, including dependency features, performs best.

engine	features	F_1	recall	precision
FSPar	F1	51.8%	56.6%	47.8%
FSPar	F1,F2	72.1%	68.9%	75.7%
FSPar	F1,F2,F3	78.3%	74.1%	83.0%
OpenNLP	F1	48.0%	62.8%	38.8%
OpenNLP	F1,F2	73.3%	71.7%	74.7%

Table 3: Results of different feature combinations on the test set for river-bodyOfWater extraction

6.4 Evaluation of self-annotation

To evaluate the quality of self-annotation, we randomly selected one set of 100 self-annotated articles from each data set and labeled these sets manually. These annotations are used to calculate the inter-annotator agreement between the human annotated and machine annotated instances. We use Cohen's κ as measure and get a result of 1.00 for the town-suburb relation. For the river-bodyOfWater relation we got a κ-value of 0.79, which also indicates good agreement.

We also use a gazetteer to evaluate the quality of all town-suburb relations that were extracted for our self-annotated training set. The accuracy is nearly perfect (only one single error), which is good evidence for the high quality of Wikipedia.

Required size of self-annotated training set. The performance of a supervised system depends on the size of the training data. In the self-annotation step a minimum of instances has to be annotated, but it is not necessary to self-annotate all available articles.

We reduced the number of articles used in the training size to test this hypothesis. Reducing the entire training set of 9000 (respectively, 3000) self-annotated articles to 1000 reduces F1 by 2.0% for town-suburb and by 2.4% for river-bodyOfWater; a reduction to 100 reduces F1 by 8.5% for town-suburb and by 9.3% for river-bodyOfWater (compared to the 9000/3000 baseline).

7 Discussion

Wu and Weld (2007) observed schema drift in their work: Wikipedia authors do not not use infobox attributes in a consistent manner. However, we did not find schema drift to be a large problem in our experiments. The variation we found can easily be handled with a small number of rules. This can be due to the fact that the quality of Wikipedia articles improved a lot in the last years through the introduction of automatic maintenance tools like bots[2]. Nevertheless, the development of self-annotation for a new relation type requires some manual work. The developer has to check the quality of the extraction relations in the infoboxes. This can lead to some additional adaptation work for the used attributes such as merging or creating rules. However, a perfect coverage is not required because the extraction system is only used for training purposes; we only need to find a sufficiently large number of positive training instances and do not require exhaustive labeling of all articles.

It is important to note that considering partially found relations as negative samples has to be avoided. Wrong negative samples have a generally unwanted impact on the performance of the learned extraction model. A developer has to be aware of this fact. In one experiment, the learned classifiers were applied to the training data and returned a number of false positive results – 40 in case of the river-bodyOfWater relation. 31 of these errors were not actual errors because the self-annotation missed some true instances. Nevertheless, the trained model recognizes these samples as correct; this could perhaps be used to further improve the quality of self-annotation.

Manually labeled data also includes noise and the benefit of self-annotation is substantial when the aim is to build a fine-grained relation extraction system in a fast and cheap way.

The difference of the results between OpenNLP and FSPar engines are smaller than expected. Although sentence splitting is poorly done by OpenNLP the effect on the extraction result is rather low. Another crucial point is that the lexicon-based named entity recognizer of the FSPar engine that was optimized for named entities used in Wikipedia has no significant impact on the overall performance. Thus, a basic set of NLP components with moderate error rates may be sufficient for effective self-annotation.

8 Conclusion

This paper described a new approach to developing and implementing a complete system to extract fine-grained geospatial relations by using a supervised machine learning approach without expensive manual labeling. Using self-annotation, systems can be rapidly developed and adapted for new relations without expensive manual annotation. Only some manual work has to be done to find the right attributes in the infoboxes. The matching process between infoboxes and text is not in all cases trivial and for some attributes additional rules have to be modeled.

9 Acknowledgment

This project was funded by DFG as part of Nexus (Collaborative Research Centre, SFB 627).

References

Auer, Sören, Christian Bizer, Georgi Kobilarov, Jens Lehmann, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *In 6th Intl Semantic Web Conference, Busan, Korea*, pages 11–15. Springer.

Blessing, Andre, Stefan Klatt, Daniela Nicklas, Steffen Volz, and Hinrich Schütze. 2006. Language-derived information and context models. In *Proceedings of 3rd IEEE PerCom Workshop on Context Modeling and Reasoning (CoMoRea) (at 4th IEEE International Conference on Pervasive Computing and Communication (PerCom'06))*.

Dürr, Frank, Nicola Hönle, Daniela Nicklas, Christian Becker, and Kurt Rothermel. 2004. Nexus–a platform for context-aware applications. In Roth, Jörg,

[2] See en.wikipedia.org/wiki/Wikipedia:Bots. The edit history of many articles shows that there is a lot of automatic maintenance by bots to avoid schema drift.

editor, *1. Fachgespräch Ortsbezogene Anwendungen und Dienste der GI-Fachgruppe KuVS*, pages 15–18, Hagen, Juni. Informatik-Bericht der FernUniversität Hagen.

Hahn, Udo, Ekaterina Buyko, Rico Landefeld, Matthias Mühlhausen, Michael Poprat, Katrin Tomanek, and Joachim Wermter. 2008. An overview of JCoRe, the JULIE lab UIMA component repository. In *Proceedings of the LREC'08 Workshop 'Towards Enhanced Interoperability for Large HLT Systems: UIMA for NLP'*, Marrakech, Morocco, May.

Jiang, Jing. 2009. Multi-task transfer learning for weakly-supervised relation extraction. In *ACL-IJCNLP '09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 1012–1020, Morristown, NJ, USA. Association for Computational Linguistics.

Müller, Christof, Torsten Zesch, Mark-Christoph Müller, Delphine Bernhard, Kateryna Ignatova, Iryna Gurevych, and Max Mühlhäuser. 2008. Flexible uima components for information retrieval research. In *Proceedings of the LREC 2008 Workshop 'Towards Enhanced Interoperability for Large HLT Systems: UIMA for NLP',* Marrakech, Morocco, May 31, 2008. 24–27.

Nothman, Joel, Tara Murphy, and James R. Curran. 2009. Analysing wikipedia and gold-standard corpora for ner training. In *EACL '09: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 612–620, Morristown, NJ, USA. Association for Computational Linguistics.

Ogren, Philip V., Philipp G. Wetzler, and Steven Bethard. 2008. Cleartk: A uima toolkit for statistical natural language processing. In *UIMA for NLP workshop at Language Resources and Evaluation Conference (LREC).*

Schiehlen, Michael. 2003. Combining deep and shallow approaches in parsing german. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 112–119, Morristown, NJ, USA. Association for Computational Linguistics.

Schmid, Helmut. 1995. Improvements in part-of-speech tagging with an application to german. In *In Proceedings of the ACL SIGDAT-Workshop*, pages 47–50.

Suchanek, Fabian M., Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *16th international World Wide Web conference (WWW 2007)*, New York, NY, USA. ACM Press.

Wu, Fei and Daniel S. Weld. 2007. Autonomously semantifying wikipedia. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, Lisbon, Portugal, November 6-10, 2007*, pages 41–50.

Zesch, Torsten, Christof Müller, and Iryna Gurevych. 2008. Extracting Lexical Semantic Knowledge from Wikipedia and Wiktionary. In *Proceedings of the Conference on Language Resources and Evaluation (LREC).*

Zhang, Ziqi and José Iria. 2009. A novel approach to automatic gazetteer generation using wikipedia. In *People's Web '09: Proceedings of the 2009 Workshop on The People's Web Meets NLP*, pages 1–9, Morristown, NJ, USA. Association for Computational Linguistics.

Very High Accuracy and Fast Dependency Parsing is not a Contradiction

Bernd Bohnet
University of Stuttgart
Institut für Maschinelle Sprachverarbeitung
bernd.bohnet@ims.uni-stuttgart.de

Abstract

In addition to a high accuracy, short parsing and training times are the most important properties of a parser. However, parsing and training times are still relatively long. To determine why, we analyzed the time usage of a dependency parser. We illustrate that the mapping of the features onto their weights in the support vector machine is the major factor in time complexity. To resolve this problem, we implemented the passive-aggressive perceptron algorithm as a Hash Kernel. The Hash Kernel substantially improves the parsing times and takes into account the features of negative examples built during the training. This has lead to a higher accuracy. We could further increase the parsing and training speed with a parallel feature extraction and a parallel parsing algorithm. We are convinced that the Hash Kernel and the parallelization can be applied successful to other NLP applications as well such as transition based dependency parsers, phrase structrue parsers, and machine translation.

1 Introduction

Highly accurate dependency parsers have high demands on resources and long parsing times. The training of a parser frequently takes several days and the parsing of a sentence can take on average up to a minute. The parsing time usage is important for many applications. For instance, dialog systems only have a few hundred milliseconds to analyze a sentence and machine translation systems, have to consider in that time some thousand translation alternatives for the translation of a sentence.

Parsing and training times can be improved by methods that maintain the accuracy level, or methods that trade accuracy against better parsing times. Software developers and researchers are usually unwilling to reduce the quality of their applications. Consequently, we have to consider at first methods to improve a parser, which do not involve an accuracy loss, such as faster algorithms, faster implementation of algorithms, parallel algorithms that use several CPU cores, and feature selection that eliminates the features that do not improve accuracy.

We employ, as a basis for our parser, the second order maximum spanning tree dependency parsing algorithm of Carreras (2007). This algorithm frequently reaches very good, or even the best labeled attachment scores, and was one of the most used parsing algorithms in the shared task 2009 of the Conference on Natural Language Learning (CoNLL) (Hajič et al., 2009). We combined this parsing algorithm with the passive-aggressive perceptron algorithm (Crammer et al., 2003; McDonald et al., 2005; Crammer et al., 2006). A parser build out of these two algorithms provides a good baseline and starting point to improve upon the parsing and training times.

The rest of the paper is structured as follows. In Section 2, we describe related work. In section 3, we analyze the time usage of the components of

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 89–97,
Beijing, August 2010

the parser. In Section 4, we introduce a new Kernel that resolves some of the bottlenecks and improves the performance. In Section 5, we describe the parallel parsing algorithms which nearly allowed us to divide the parsing times by the number of cores. In Section 6, we determine the optimal setting for the Non-Projective Approximation Algorithm. In Section 7, we conclude with a summary and an outline of further research.

2 Related Work

The two main approaches to dependency parsing are transition based dependency parsing (Nivre, 2003; Yamada and Matsumoto., 2003; Titov and Henderson, 2007) and maximum spanning tree based dependency parsing (Eisner, 1996; Eisner, 2000; McDonald and Pereira, 2006). Transition based parsers typically have a linear or quadratic complexity (Nivre et al., 2004; Attardi, 2006). Nivre (2009) introduced a transition based non-projective parsing algorithm that has a worst case quadratic complexity and an expected linear parsing time. Titov and Henderson (2007) combined a transition based parsing algorithm, which used a beam search with a latent variable machine learning technique.

Maximum spanning tree dependency based parsers decomposes a dependency structure into parts known as “factors”. The factors of the first order maximum spanning tree parsing algorithm are edges consisting of the head, the dependent (child) and the edge label. This algorithm has a quadratic complexity. The second order parsing algorithm of McDonald and Pereira (2006) uses a separate algorithm for edge labeling. This algorithm uses in addition to the first order factors: the edges to those children which are closest to the dependent. The second order algorithm of Carreras (2007) uses in addition to McDonald and Pereira (2006) the child of the dependent occurring in the sentence between the head and the dependent, and the an edge to a grandchild. The edge labeling is an integral part of the algorithm which requires an additional loop over the labels. This algorithm therefore has a complexity of O(n^4). Johansson and Nugues (2008) reduced the needed number of loops over the edge labels by using only the edges that existed in the training corpus for a distinct head and child part-of-speech tag combination.

The transition based parsers have a lower complexity. Nevertheless, the reported run times in the last shared tasks were similar to the maximum spanning tree parsers. For a transition based parser, Gesmundo et al. (2009) reported run times between 2.2 days for English and 4.7 days for Czech for the joint training of syntactic and semantic dependencies. The parsing times were about one word per second, which speeds up quickly with a smaller beam-size, although the accuracy of the parser degrades a bit. Johansson and Nugues (2008) reported training times of 2.4 days for English with the high-order parsing algorithm of Carreras (2007).

3 Analysis of Time Usage

We built a baseline parser to measure the time usage. The baseline parser resembles the architecture of McDonald and Pereira (2006). It consists of the second order parsing algorithm of Carreras (2007), the non-projective approximation algorithm (McDonald and Pereira, 2006), the passive-aggressive support vector machine, and a feature extraction component. The features are listed in Table 4. As in McDonald et al. (2005), the parser stores the features of each training example in a file. In each epoch of the training, the feature file is read, and the weights are calculated and stored in an array. This procedure is up to 5 times faster than computing the features each time anew. But the parser has to maintain large arrays: for the weights of the sentence and the training file. Therefore, the parser needs 3GB of main memory for English and 100GB of disc space for the training file. The parsing time is approximately 20% faster, since some of the values did not have to be recalculated.

Algorithm 1 illustrates the training algorithm in pseudo code. τ is the set of training examples where an example is a pair (x_i, y_i) of a sentence and the corresponding dependency structure. $\overrightarrow{w}$ and $\overrightarrow{v}$ are weight vectors. The first loop extracts features from the sentence x_i and maps the features to numbers. The numbers are grouped into three vectors for the features of all possible edges $\phi_{h,d}$, possible edges in combination with siblings $\phi_{h,d,s}$ and in combination with grandchil-

	t_{e+s}	t_r	t_p	t_a	rest	total	t_e	pars.	train.	sent.	feat.	LAS	UAS
Chinese	4582	748	95	-	3	846	3298	3262	84h	22277	8.76M	76.88	81.27
English	1509	168	12.5	20	1.5	202	1223	1258	38.5h	39279	8.47M	90.14	92.45
German	945	139	7.7	17.8	1.5	166	419	429	26.7h	36020	9.16M	87.64	90.03
Spanish	3329	779	36	-	2	816	2518	2550	16.9h	14329	5.51M	86.02	89.54

Table 1: t_{e+s} is the elapsed time in milliseconds to extract and store the features, t_r to read the features and to calculate the weight arrays, t_p to predict the projective parse tree, t_a to apply the non-projective approximation algorithm, *rest* is the time to conduct the other parts such as the update function, *train.* is the total training time per instance ($t_r + t_p + t_a$+*rest*), and t_e is the elapsed time to extract the features. The next columns illustrate the parsing time in milliseconds per sentence for the test set, training time in hours, the number of sentences in the training set, the total number of features in million, the labeled attachment score of the test set, and the unlabeled attachment score.

Algorithm 1: Training – baseline algorithm

$\tau = \{(x_i, y_i)\}_{i=1}^{I}$ // Training data
$\overrightarrow{w} = 0, \overrightarrow{v} = 0$
$\gamma = E * I$ // passive-aggresive update weight
for i = 1 **to** I
  t^s_{s+e}; extract-and-store-features(x_i); t^e_{s+e};
for n = 1 **to** E // iteration over the training epochs
  for i = 1 **to** I // iteration over the training examples
    $k \leftarrow (n-1) * I + i$
    $\gamma = E * I - k + 2$ // passive-aggressive weight
    $t^s_{r,k}$; A = read-features-and-calc-arrays(i,$\overrightarrow{w}$) ; $t^e_{r,k}$
    $t^s_{p,k}$; y_p = predicte-projective-parse-tree(A);$t^e_{p,k}$
    $t^s_{a,k}$; y_a = non-projective-approx.(y_p,A); $t^e_{a,k}$
    update $\overrightarrow{w}$, $\overrightarrow{v}$ according to $\Delta(y_p, y_i)$ and γ
$w = v/(E * I)$ // average

dren $\phi_{h,d,g}$ where h, d, g, and s are the indexes of the words included in x_i. Finally, the method stores the feature vectors on the hard disc.

The next two loops build the main part of the training algorithm. The outer loop iterates over the number of training epochs, while the inner loop iterates over all training examples. The online training algorithm considers a single training example in each iteration. The first function in the loop reads the features and computes the weights A for the factors in the sentence x_i. A is a set of weight arrays.

$$A = \{\overrightarrow{w} * \overrightarrow{f}_{h,d}, \overrightarrow{w} * \overrightarrow{f}_{h,d,s}, \overrightarrow{w} * \overrightarrow{f}_{h,d,g}\}$$

The parsing algorithm uses the weight arrays to predict a projective dependency structure y_p. The non-projective approximation algorithm has as input the dependency structure and the weight arrays. It rearranges the edges and tries to increase the total score of the dependency structure. This algorithm builds a dependency structure y_a, which might be non-projective. The training algorithm updates $\overrightarrow{w}$ according to the difference between the predicted dependency structures y_a and the reference structure y_i. It updates $\overrightarrow{v}$ as well, whereby the algorithm additionally weights the updates by γ. Since the algorithm decreases γ in each round, the algorithm adapts the weights more aggressively at the beginning (Crammer et al., 2006). After all iterations, the algorithm computes the average of $\overrightarrow{v}$, which reduces the effect of overfitting (Collins, 2002).

We have inserted into the training algorithm functions to measure the start times t^s and the end times t^e for the procedures to compute and store the features, to read the features, to predict the projective parse, and to calculate the non-projective approximation. We calculate the average elapsed time per instance, as the average over all training examples and epochs:

$$t_x = \frac{\sum_{k=1}^{E*I} t^e_{x,k} - t^s_{x,k}}{E*I}.$$

We use the training set and the test set of the CoNLL shared task 2009 for our experiments. Table 1 shows the elapsed times in $\frac{1}{1000}$ seconds (milliseconds) of the selected languages for the procedure calls in the loops of Algorithm 1. We had to measure the times for the feature extraction in the parsing algorithm, since in the training algorithm, the time can only be measured together with the time for storing the features. The table contains additional figures for the total training time and parsing scores.[1]

The parsing algorithm itself only required, to our surprise, 12.5 ms (t_p) for a English sentence

[1] We use a Intel Nehalem i7 CPU 3.33 Ghz. With turbo mode on, the clock speed was 3.46 Ghz.

on average, while the feature extraction needs 1223 ms. To extract the features takes about 100 times longer than to build a projective dependency tree. The feature extraction is already implemented efficiently. It uses only numbers to represent features which it combines to a long integer number and then maps by a hash table[2] to a 32bit integer number. The parsing algorithm uses the integer number as an index to access the weights in the vectors $\overrightarrow{w}$ and $\overrightarrow{v}$.

The complexity of the parsing algorithm is usually considered the reason for long parsing times. However, it is not the most time consuming component as proven by the above analysis. Therefore, we investigated the question further, asking what causes the high time consumption of the feature extraction?

In our next experiment, we left out the mapping of the features to the index of the weight vectors. The feature extraction takes 88 ms/sentence without the mapping and 1223 ms/sentence with the mapping. The feature–index mapping needs 93% of the time to extract the features and 91% of the total parsing time. What causes the high time consumption of the feature–index mapping?

The mapping has to provide a number as an index for the features in the training examples and to filter out the features of examples built, while the parser predicts the dependency structures. The algorithm filters out negative features to reduce the memory requirement, even if they could improve the parsing result. We will call the features built due to the training examples positive features and the rest negative features. We counted 5.8 times more access to negative features than positive features.

We now look more into the implementation details of the used hash table to answer the previously asked question. The hash table for the feature–index mapping uses three arrays: one for the keys, one for the values and a status array to indicate the deleted elements. If a program stores a value then the hash function uses the key to calculate the location of the value. Since the hash function is a heuristic function, the predicted location might be wrong, which leads to so-called hash misses. In such cases the hash algorithm has to retry to find the value. We counted 87% hash misses including misses where the hash had to retry several times. The number of hash misses was high, because of the additional negative features. The CPU cache can only store a small amount of the data from the hash table. Therefore, the memory controller has frequently to transfer data from the main memory into the CPU. This procedure is relatively slow. We traced down the high time consumption to the access of the key and the access of the value. Successive accesses to the arrays are fast, but the relative random accesses via the hash function are very slow. The large number of accesses to the three arrays, because of the negative features, positive features and because of the hash misses multiplied by the time needed to transfer the data into the CPU are the reason for the high time consumption.

We tried to solve this problem with Bloom filters, larger hash tables and customized hash functions to reduce the hash misses. These techniques did not help much. However, a substantial improvement did result when we eliminated the hash table completely, and directly accessed the weight vectors $\overrightarrow{w}$ and $\overrightarrow{v}$ with a hash function. This led us to the use of Hash Kernels.

4 Hash Kernel

A Hash Kernel for structured data uses a hash function $h : J \rightarrow \{1...n\}$ to index ϕ, cf. Shi et al. (2009). ϕ maps the observations X to a feature space. We define $\phi(x, y)$ as the numeric feature representation indexed by J. Let $\overline{\phi}_k(x, y) = \phi_j(x, y)$ the hash based feature–index mapping, where $h(j) = k$. The process of parsing a sentence x_i is to find a parse tree y_p that maximizes a scoring function $\text{argmax}_y F(x_i, y)$. The learning problem is to fit the function F so that the errors of the predicted parse tree y are as low as possible. The scoring function of the Hash Kernel is

$$F(x, y) = \overrightarrow{w} * \overline{\phi}(x, y)$$

where $\overrightarrow{w}$ is the weight vector and the size of $\overrightarrow{w}$ is n.

Algorithm 2 shows the update function of the Hash Kernel. We derived the update function from the update function of MIRA (Crammer et

[2]We use the hash tables of the *trove* library: `http://sourceforge.net/projects/trove4j`.

Algorithm 2: Update of the Hash Kernel

// y_p = arg max$_y F(x_i, y)$
update($\overrightarrow{w}, \overrightarrow{v}, x_i, y_i, y_p, \gamma$**)**
  $\epsilon = \Delta(y_i, y_p)$ // number of wrong labeled edges
  if $\epsilon > 0$ **then**
    $\overrightarrow{u} \leftarrow (\overline{\phi}(x_i, y_i) - \overline{\phi}(x_i, y_p))$
    $\nu = \frac{\epsilon - (F(x_t, y_i) - F(x_i, y_p))}{||\overrightarrow{u}||^2}$
    $\overrightarrow{w} \leftarrow \overrightarrow{w} + \nu * \overrightarrow{u}$
    $\overrightarrow{v} \leftarrow \vec{v} + \gamma * \nu * \overrightarrow{u}$
return $\overrightarrow{w}, \overrightarrow{v}$

al., 2006). The parameters of the function are the weight vectors $\overrightarrow{w}$ and $\overrightarrow{v}$, the sentence x_i, the gold dependency structure y_i, the predicted dependency structure y_p, and the update weight γ. The function Δ calculates the number of wrong labeled edges. The update function updates the weight vectors, if at least one edge is labeled wrong. It calculates the difference $\overrightarrow{u}$ of the feature vectors of the gold dependency structure $\overline{\phi}(x_i, y_i)$ and the predicted dependency structure $\overline{\phi}(x_i, y_p)$. Each time, we use the feature representation ϕ, the hash function h maps the features to integer numbers between 1 and $|\overrightarrow{w}|$. After that the update function calculates the margin ν and updates $\overrightarrow{w}$ and $\overrightarrow{v}$ respectively.

Algorithm 3 shows the training algorithm for the Hash Kernel in pseudo code. A main difference to the baseline algorithm is that it does not store the features because of the required time which is needed to store the additional negative features. Accordingly, the algorithm first extracts the features for each training instance, then maps the features to indexes for the weight vector with the hash function and calculates the weight arrays.

Algorithm 3: Training – Hash Kernel

for n $\leftarrow$ 1 **to** E // iteration over the training epochs
  for i $\leftarrow$ 1 **to** I // iteration over the training exmaples
    $k \leftarrow (n-1) * I + i$
    $\gamma \leftarrow E * I - k + 2$ // passive-aggressive weight
    $t^s_{e,k}$; $A \leftarrow$ extr.-features-&-calc-arrays(i,$\overrightarrow{w}$) ; $t^e_{e,k}$
    $t^s_{p,k}$; $y_p \leftarrow$ predicte-projective-parse-tree(A);$t^e_{p,k}$
    $t^s_{a,k}$; $y_a \leftarrow$ non-projective-approx.(y_p,A); $t^e_{a,k}$
    update $\overrightarrow{w}, \overrightarrow{v}$ according to $\Delta(y_p, y_i)$ and γ
$w = v/(E * I)$ // average

For different j, the hash function $h(j)$ might generate the same value k. This means that the hash function maps more than one feature to the same weight. We call such cases collisions. Collisions can reduce the accuracy, since the weights are changed arbitrarily. This procedure is similar to randomization of weights (features), which aims to save space by sharing values in the weight vector (Blum., 2006; Rahimi and Recht, 2008). The Hash Kernel shares values when collisions occur that can be considered as an approximation of the kernel function, because a weight might be adapted due to more than one feature. If the approximation works well then we would need only a relatively small weight vector otherwise we need a larger weight vector to reduce the chance of collisions. In an experiments, we compared two hash functions and different hash sizes. We selected for the comparison a standard hash function (h_1) and a custom hash function (h_2). The idea for the custom hash function h_2 is not to overlap the values of the feature sequence number and the edge label with other values. These values are stored at the beginning of a long number, which represents a feature.

$$h_1 \leftarrow |(l\ xor(l \vee \text{0xffffffff00000000} >> 32))\%\ \text{size}|^3$$

$$\begin{aligned} h_2 \leftarrow |(l\ xor\ & ((l >> 13)\ \vee\ \text{0xffffffffffffe000})\ xor \\ & ((l >> 24)\ \vee\ \text{0xffffffffffff0000})\ xor \\ & ((l >> 33)\ \vee\ \text{0xfffffffffffc0000})\ xor \\ & ((l >> 40)\ \vee\ \text{0xfffffffffff00000}))\ \%\ \text{size}\ | \end{aligned}$$

vector size	h_1	#(h_1)	h_2	#(h_2)
411527	85.67	0.41	85.74	0.41
3292489	87.82	3.27	87.97	3.28
10503061	88.26	8.83	88.35	8.77
21006137	88.19	12.58	88.41	12.53
42012281	88.32	12.45	88.34	15.27
115911564*	88.32	17.58	88.39	17.34
179669557	88.34	17.65	88.28	17.84

Table 2: The labeled attachment scores for different weight vector sizes and the number of nonzero values in the feature vectors in millions. * Not a prime number.

Table 2 shows the labeled attachment scores for selected weight vector sizes and the number of nonzero weights. Most of the numbers in Table 2 are primes, since they are frequently used to obtain a better distribution of the content in hash ta-

[3] $>> n$ shifts n bits right, and $\%$ is the modulo operation.

bles. h_2 has more nonzero weights than h_1. Nevertheless, we did not observe any clear improvement of the accuracy scores. The values do not change significantly for a weight vector size of 10 million and more elements. We choose a weight vector size of 115911564 values for further experiments since we get more non zero weights and therefore fewer collisions.

	t_e	t_p	t_a	r	total	par.	trai.
Chinese	1308	-	200	3	1511	1184	93h
English	379	21.3	18.2	1.5	420	354	46h
German	209	12	15.3	1.7	238	126	24h
Spanish	1056	-	39	2	1097	1044	44h

Table 3: The time in milliseconds for the feature extraction, projective parsing, non-projective approximation, rest (r), the total training time per instance, the average parsing (par.) time in milliseconds for the test set and the training time in hours

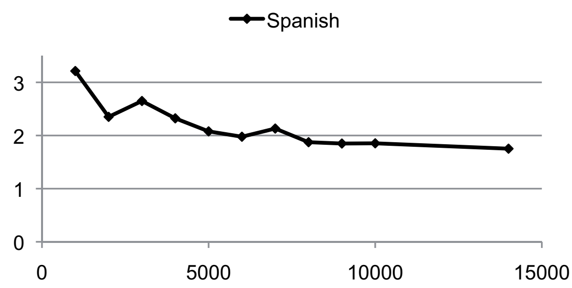


Figure 1: The difference of the labeled attachment score between the baseline parser and the parser with the Hash Kernel (y-axis) for increasing large training sets (x-axis).

Table 3 contains the measured times for the Hash Kernel as used in Algorithm 2. The parser needs 0.354 seconds in average to parse a sentence of the English test set. This is 3.5 times faster than the baseline parser. The reason for that is the faster feature mapping of the Hash Kernel. Therefore, the measured time t_e for the feature extraction and the calculation of the weight arrays are much lower than for the baseline parser. The training is about 19% slower since we could no longer use a file to store the feature indexes of the training examples because of the large number of negative features. We counted about twice the number of nonzero weights in the weight vector of the Hash Kernel compared to the baseline parser. For instance, we counted for English 17.34 Millions nonzero weights in the Hash Kernel and 8.47 Millions in baseline parser and for Chinese 18.28 Millions nonzero weights in the Hash Kernel and 8.76 Millions in the baseline parser. Table 6 shows the scores for all languages of the shared task 2009. The attachment scores increased for all languages. It increased most for Catalan and Spanish. These two corpora have the smallest training sets. We searched for the reason and found that the Hash Kernel provides an overproportional accuracy gain with less training data compared to MIRA. Figure 1 shows the difference between the labeled attachment score of the parser with MIRA and the Hash Kernel for Spanish. The decreasing curve shows clearly that the Hash Kernel provides an overproportional accuracy gain with less training data compared to the baseline. This provides an advantage for small training corpora.

However, this is probably not the main reason for the high improvement, since for languages with only slightly larger training sets such as Chinese the improvement is much lower and the gradient at the end of the curve is so that a huge amount of training data would be needed to make the curve reach zero.

5 Parallelization

Current CPUs have up to 12 cores and we will see soon CPUs with more cores. Also graphic cards provide many simple cores. Parsing algorithms can use several cores. Especially, the tasks to extract the features and to calculate the weight arrays can be well implemented as parallel algorithm. We could also successful parallelize the projective parsing and the non-projective approximation algorithm. Algorithm 4 shows the parallel feature extraction in pseudo code. The main method prepares a list of tasks which can be performed in parallel and afterwards it creates the threads that perform the tasks. Each thread removes from the task list an element, carries out the task and stores the result. This procedure is repeated until the list is empty. The main method waits until all threads are completed and returns the result. For the parallel algorithms, Table 5 shows the elapsed times depend on the number of

#	**Standard Features**	#	**Linear Features**		**Linear G. Features**		**Sibling Features**
1	$l,h_f,h_p,d(h,d)$	14	$l,h_p,h+1_p,d_p,d(h,d)$	44	$l,g_p,d_p,d+1_p,d(h,d)$	99	$l,s_l,h_p,d(h,d)\oplus r(h,d)$
2	$l,h_f,d(h,d)$	15	$l,h_p,d-1_p,d_p,d(h,d)$	45	$l,g_p,d_p,d-1_p,d(h,d)$	100	$l,s_l,d_p,d(h,d)\oplus r(h,d)$
3	$l,h_p,d(h,d)$	16	$l,h_p,d_p,d+1_p,d(h,d)$	46	$l,g_p,g+1_p,d-1_p,d_p,d(h,d)$	101	$l,h_l,d_p,d(h,d)\oplus r(h,d)$
4	$l,d_f,d_p,d(h,d)$	17	$l,h_p,h+1_p,d-1_p,d_p,d(h,d)$	47	$l,g-1_p,g_p,d-1_p,d_p,d(h,d)$	102	$l,d_l,s_p,d(h,d)\oplus r(h,d)$
5	$l,h_p,d(h,d)$	18	$l,h-1_p,h+1_p,d-1_p,d_p,d(h,d)$	48	$l,g_p,g+1_p,d_p,d+1_p,d(h,d)$	75	$l,\forall d_m,\forall s_m,d(h,d)$
6	$l,d_p,d(h,d)$	19	$l,h_p,h+1_p,d_p,d+1_p,d(h,d)$	49	$l,g-1_p,g_p,d_p,d+1_p,d(h,d)$	76	$l,\forall h_m,\forall s_m,d(h,s)$
7	$l,h_f,h_p,d_f,d_p,d(h,d)$	20	$l,h-1_p,h_p,d_p,d-1_p,d(h,d)$	50	$l,g_p,g+1_p,h_p,d(h,d)$		**Linear S. Features**
8	$l,h_p,d_f,d_p,d(h,d)$		**Grandchild Features**	51	$l,g_p,g-1_p,h_p,d(h,d)$	58	$l,s_p,s+1_p,h_p,d(h,d)$
9	$l,h_f,d_f,d_p,d(h,d)$	21	$l,h_p,d_p,g_p,d(h,d,g)$	52	$l,g_p,h_p,h+1_p,d(h,d)$	59	$l,s_p,s-1_p,h_p,d(h,d)$
10	$l,h_f,h_p,d_f,d(h,d)$	22	$l,h_p,g_p,d(h,d,g)$	53	$l,g_p,h_p,h-1_p,d(h,d)$	60	$l,s_p,h_p,h+1_p,d(h,d)$
11	$l,h_f,d_f,h_p,d(h,d)$	23	$l,d_p,g_p,d(h,d,g)$	54	$l,g_p,g+1_p,h-1_p,h_p,d(h,d)$	61	$l,s_p,h_p,h-1_p,d(h,d)$
12	$l,h_f,d_f,d(h,d)$	24	$l,h_f,g_f,d(h,d,g)$	55	$l,g-1_p,g_p,h-1_p,h_p,d(h,d)$	62	$l,s_p,s+1_p,h-1_p,d(h,d)$
13	$l,h_p,d_p,d(h,d)$	25	$l,d_f,g_f,d(h,d,g)$	56	$l,g_p,g+1_p,h_p,h+1_p,d(h,d)$	63	$l,s-1_p,s_p,h-1_p,d(h,d)$
77	$l,h_l,h_p,d(h,d)$	26	$l,g_f,h_p,d(h,d,g)$	57	$l,g-1_p,g_p,h_p,h+1_p,d(h,d)$	64	$l,s_p,s+1_p,h_p,d(h,d)$
78	$l,h_l,d(h,d)$	27	$l,g_f,d_p,d(h,d,g)$		**Sibling Features**	65	$l,s-1_p,s_p,h_p,h+1_p,d(h,d)$
79	$l,h_p,d(h,d)$	28	$l,h_f,g_p,d(h,d,g)$	30	$l,h_p,d_p,s_p,d(h,d)\oplus r(h,d)$	66	$l,s_p,s+1_p,d_p,d(h,d)$
80	$l,d_l,d_p,d(h,d)$	29	$l,d_f,g_p,d(h,d,g)$	31	$l,h_p,s_p,d(h,d)\oplus r(h,d)$	67	$l,s_p,s-1_p,d_p,d(h,d)$
81	$l,d_l,d(h,d)$	91	$l,h_l,g_l,d(h,d,g)$	32	$l,d_p,s_p,d(h,d)\oplus r(h,d)$	68	$s_p,d_p,d+1_p,d(h,d)$
82	$l,d_p,d(h,d)$	92	$l,d_p,g_p,d(h,d,g)$	33	$l,p_f,s_f,d(h,d)\oplus r(h,d)$	69	$s_p,d_p,d-1_p,d(h,d)$
83	$l,d_l,h_p,d_p,h_l,d(h,d)$	93	$l,g_l,h_p,d(h,d,g)$	34	$l,p_p,s_f,d(h,d)\oplus r(h,d)$	70	$s_p,s+1_p,d-1_p,d_p,d(h,d)$
84	$l,d_l,h_p,d_p,d(h,d)$	94	$l,g_l,d_p,d(h,d,g)$	35	$l,s_f,p_p,d(h,d)\oplus r(h,d)$	71	$s-1_p,s_p,d-1_p,d_p,d(h,d)$
85	$l,h_l,d_l,d_p,d(h,d)$	95	$l,h_l,g_p,d(h,d,g)$	36	$l,s_f,d_p,d(h,d)\oplus r(h,d)$	72	$s_p,s+1_p,d_p,d+1_p,d(h,d)$
86	$l,h_l,h_p,d_p,d(h,d)$	96	$l,d_l,g_p,d(h,d,g)$	37	$l,s_f,d_p,d(h,d)\oplus r(h,d)$	73	$s-1_p,s_p,d_p,d+1_p,d(h,d)$
87	$l,h_l,d_l,h_p,d(h,d)$	74	$l,\forall d_m,\forall g_m,d(h,d)$	38	$l,d_f,s_p,d(h,d)\oplus r(h,d)$		**Special Feature**
88	$l,h_l,d_l,d(h,d)$		**Linear G. Features**	97	$l,h_l,s_l,d(h,d)\oplus r(h,d)$	39	$\forall l,h_p,d_p,x_p$ between h,d
89	$l,h_p,d_p,d(h,d)$	42	$l,g_p,g+1_p,d_p,d(h,d)$	98	$l,d_l,s_l,d(h,d)\oplus r(h,d)$		
41	$l,\forall h_m,\forall d_m,d(h,d)$	43	$l,g_p,g-1_p,d_p,d(h,d)$				

Table 4: Features Groups. *l* represents the label, *h* the head, d the dependent, *s* a sibling, and *g* a grandchild, **d**(x,y,[,z]) the order of words, and **r**(x,y) the distance.

used cores. The parsing time is 1.9 times faster on two cores and 3.4 times faster on 4 cores. Hyper threading can improve the parsing times again and we get with hyper threading 4.6 faster parsing times. Hyper threading possibly reduces the overhead of threads, which contains already our single core version.

Algorithm 4: Parallel Feature Extraction

A // weight arrays
extract-features-and-calc-arrays(x_i)
  data-list ← {} // thread-save data list
  for $w_1 \leftarrow 1$ to $|x_i|$
    for $w_2 \leftarrow 1$ to $|x_i|$
      data-list ← data-list $\cup\{(w_1, w_2)\}$
  c ← number of CPU cores
  for $t \leftarrow 1$ to c
    T_t ← create-array-thread(t, x_i,data-list)
    start array-thread T_t// start thread t
  for $t \leftarrow 1$ to c
    join T_t// wait until thread t is finished
    $A \leftarrow\ A \cup$ collect-result(T_t)
return A
//
array-thread T
  d ← remove-first-element(data-list)
  if d is empty **then** end-thread
  ... // extract features and calculate part d of A

Cores	t_e	t_p	t_a	rest	total	pars.	train.
1	379	21.3	18.2	1.5	420	354	45.8h
2	196	11.7	9.2	2.1	219	187	23.9h
3	138	8.9	6.5	1.6	155	126	16.6h
4	106	8.2	5.2	1.6	121	105	13.2h
4+4h	73.3	8.8	4.8	1.3	88.2	77	9.6h

Table 5: Elapsed times in milliseconds for different numbers of cores. The parsing time (pars.) are expressed in milliseconds per sentence and the training (train.) time in hours. The last row shows the times for 8 threads on a 4 core CPU with Hyper-threading. For these experiment, we set the clock speed to 3.46 Ghz in order to have the same clock speed for all experiments.

6 Non-Projective Approximation Threshold

For non-projective parsing, we use the Non-Projective Approximation Algorithm of McDonald and Pereira (2006). The algorithm rearranges edges in a dependency tree when they improve the score. Bohnet (2009) extended the algorithm by a threshold which biases the rearrangement of the edges. With a threshold, it is possible to gain a higher percentage of correct dependency links. We determined a threshold in experiments for Czech, English and German. In the experiment, we use the Hash Kernel and increase the thresh-

System	Average	Catalan	Chinese	Czech	English	German	Japanese	Spanish
Top CoNLL 09	85.77[(1)]	**87.86**[(1)]	**79.19**[(4)]	80.38[(1)]	89.88[(2)]	87.48[(2)]	**92.57**[(3)]	87.64[(1)]
Baseline Parser	85.10	85.70	76.88	76.93	90.14	87.64	92.26	86.12
this work	**86.33**	87.45	76.99	**80.96**	**90.33**	**88.06**	92.47	**88.13**

Table 6: Top LAS of the CoNLL 2009 of (1) Gesmundo et al. (2009), (2) Bohnet (2009), (3) Che et al. (2009), and (4) Ren et al. (2009); LAS of the baseline parser and the parser with Hash Kernel. The numbers in bold face mark the top scores. We used for Catalan, Chinese, Japanese and Spanish the projective parsing algorithm.

old at the beginning in small steps by 0.1 and later in larger steps by 0.5 and 1.0. Figure 2 shows the labeled attachment scores for the Czech, English and German development set in relation to the rearrangement threshold. The curves for all languages are a bit volatile. The English curve is rather flat. It increases a bit until about 0.3 and remains relative stable before it slightly decreases. The labeled attachment score for German and Czech increases until 0.3 as well and then both scores start to decrease. For English a threshold between 0.3 and about 2.0 would work well. For German and Czech, a threshold of about 0.3 is the best choice. We selected for all three languages a threshold of 0.3.

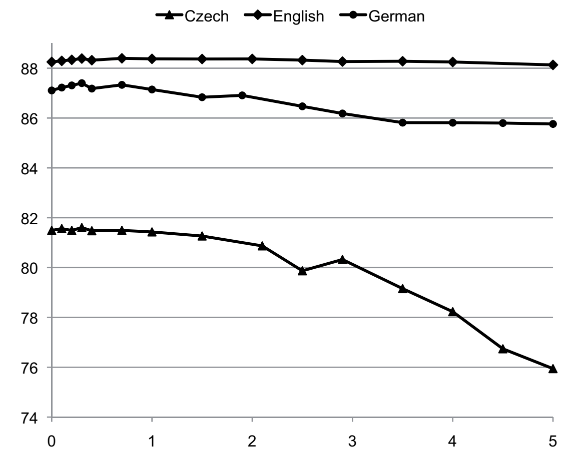


Figure 2: English, German, and Czech labeled attachment score (y-axis) for the development set in relation to the rearrangement threshold (x-axis).

7 Conclusion and Future Work

We have developed a very fast parser with excellent attachment scores. For the languages of the 2009 CoNLL Shared Task, the parser could reach higher accuracy scores on average than the top performing systems. The scores for Catalan, Chinese and Japanese are still lower than the top scores. However, the parser would have ranked second for these languages. For Catalan and Chinese, the top results obtained transition-based parsers. Therefore, the integration of both techniques as in Nivre and McDonald (2008) seems to be very promising. For instance, to improve the accuracy further, more global constrains capturing the subcategorization correct could be integrated as in Riedel and Clarke (2006). Our faster algorithms may make it feasible to consider further higher order factors.

In this paper, we have investigated possibilities for increasing parsing speed without any accuracy loss. The parsing time is 3.5 times faster on a single CPU core than the baseline parser which has an typical architecture for a maximum spanning tree parser. The improvement is due solely to the Hash Kernel. The Hash Kernel was also a prerequisite for the parallelization of the parser because it requires much less memory bandwidth which is nowadays a bottleneck of parsers and many other applications.

By using parallel algorithms, we could further increase the parsing time by a factor of 3.4 on a 4 core CPU and including hyper threading by a factor of 4.6. The parsing speed is 16 times faster for the English test set than the conventional approach. The parser needs only 77 millisecond in average to parse a sentence and the speed will scale with the number of cores that become available in future. To gain even faster parsing times, it may be possible to trade accuracy against speed. In a pilot experiment, we have shown that it is possible to reduce the parsing time in this way to as little as 9 milliseconds. We are convinced that the Hash Kernel can be applied successful to transition based dependency parsers, phrase structure parsers and many other NLP applications. [4]

[4]We provide the Parser and Hash Kernel as open source for download from http://code.google.com/p/mate-tools.

References

Attardi, G. 2006. Experiments with a Multilanguage Non-Projective Dependency Parser. In *Proceedings of CoNLL*, pages 166–170.

Blum., A. 2006. Random Projection, Margins, Kernels, and Feature-Selection. In *LNCS*, pages 52–68. Springer.

Bohnet, B. 2009. Efficient Parsing of Syntactic and Semantic Dependency Structures. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009)*.

Carreras, X. 2007. Experiments with a Higher-order Projective Dependency Parser. In *EMNLP/CoNLL*.

Che, W., Li Z., Li Y., Guo Y., Qin B., and Liu T. 2009. Multilingual Dependency-based Syntactic and Semantic Parsing. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009)*.

Collins, M. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *EMNLP*.

Crammer, K., O. Dekel, S. Shalev-Shwartz, and Y. Singer. 2003. Online Passive-Aggressive Algorithms. In *Sixteenth Annual Conference on Neural Information Processing Systems (NIPS)*.

Crammer, K., O. Dekel, S. Shalev-Shwartz, and Y. Singer. 2006. Online Passive-Aggressive Algorithms. *Journal of Machine Learning Research*, 7:551–585.

Eisner, J. 1996. Three New Probabilistic Models for Dependency Parsing: An Exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 340–345, Copenhaen.

Eisner, J., 2000. *Bilexical Grammars and their Cubic-time Parsing Algorithms*, pages 29–62. Kluwer Academic Publishers.

Gesmundo, A., J. Henderson, P. Merlo, and I. Titov. 2009. A Latent Variable Model of Synchronous Syntactic-Semantic Parsing for Multiple Languages. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009)*, Boulder, Colorado, USA., June 4-5.

Hajič, J., M. Ciaramita, R. Johansson, D. Kawahara, M. Antònia Martí, L. Màrquez, A. Meyers, J. Nivre, S. Padó, J. Štěpánek, P. Straňák, M. Surdeanu, N. Xue, and Y. Zhang. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In *Proceedings of the 13th CoNLL-2009, June 4-5*, Boulder, Colorado, USA.

Johansson, R. and P. Nugues. 2008. Dependency-based Syntactic–Semantic Analysis with PropBank and NomBank. In *Proceedings of the Shared Task Session of CoNLL-2008*, Manchester, UK.

McDonald, R. and F. Pereira. 2006. Online Learning of Approximate Dependency Parsing Algorithms. In *In Proc. of EACL*, pages 81–88.

McDonald, R., K. Crammer, and F. Pereira. 2005. Online Large-margin Training of Dependency Parsers. In *Proc. ACL*, pages 91–98.

Nivre, J. and R. McDonald. 2008. Integrating Graph-Based and Transition-Based Dependency Parsers. In *ACL-08*, pages 950–958, Columbus, Ohio.

Nivre, J., J. Hall, and J. Nilsson. 2004. Memory-Based Dependency Parsing. In *Proceedings of the 8th CoNLL*, pages 49–56, Boston, Massachusetts.

Nivre, J. 2003. An Efficient Algorithm for Projective Dependency Parsing. In *8th International Workshop on Parsing Technologies*, pages 149–160, Nancy, France.

Nivre, J. 2009. Non-Projective Dependency Parsing in Expected Linear Time. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pages 351–359, Suntec, Singapore.

Rahimi, A. and B. Recht. 2008. Random Features for Large-Scale Kernel Machines. In Platt, J.C., D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. MIT Press, Cambridge, MA.

Ren, H., D. Ji Jing Wan, and M. Zhang. 2009. Parsing Syntactic and Semantic Dependencies for Multiple Languages with a Pipeline Approach. In *Proceedings of the 13th Conference on Computational Natural Language Learning (CoNLL-2009)*, Boulder, Colorado, USA., June 4-5.

Riedel, S. and J. Clarke. 2006. Incremental Integer Linear Programming for Non-projective Dependency Parsing. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 129–137, Sydney, Australia, July. Association for Computational Linguistics.

Shi, Q., J. Petterson, G. Dror, J. Langford, A. Smola, and S.V.N. Vishwanathan. 2009. Hash Kernels for Structured Data. In *Journal of Machine Learning*.

Titov, I. and J. Henderson. 2007. A Latent Variable Model for Generative Dependency Parsing. In *Proceedings of IWPT*, pages 144–155.

Yamada, H. and Y. Matsumoto. 2003. Statistical Dependency Analysis with Support Vector Machines. In *Proceedings of IWPT*, pages 195–206.

Broad Coverage Multilingual Deep Sentence Generation with a Stochastic Multi-Level Realizer

Bernd Bohnet[1], **Leo Wanner**[1,2], **Simon Mille**[1], **Alicia Burga**[1]
[1]Department of Information and Communication Technologies
Pompeu Fabra University
[2]Institució Catalana de Recerca i Estudis Avançats (ICREA)
`{first-name.last-name}@upf.edu`

Abstract

Most of the known stochastic sentence generators use syntactically annotated corpora, performing the projection to the surface in one stage. However, in full-fledged text generation, sentence realization usually starts from semantic (predicate-argument) structures. To be able to deal with semantic structures, stochastic generators require semantically annotated, or, even better, multilevel annotated corpora. Only then can they deal with such crucial generation issues as sentence planning, linearization and morphologization. Multilevel annotated corpora are increasingly available for multiple languages. We take advantage of them and propose a multilingual deep stochastic sentence realizer that mirrors the state-of-the-art research in semantic parsing. The realizer uses an SVM learning algorithm. For each pair of adjacent levels of annotation, a separate decoder is defined. So far, we evaluated the realizer for Chinese, English, German, and Spanish.

1 Introduction

Recent years saw a significant increase of interest in corpus-based natural language generation (NLG), and, in particular, in corpus-based (or stochastic) *sentence realization*, i.e., that part of NLG which deals with mapping of a formal (more or less abstract) sentence plan onto a chain of inflected words; cf., among others, (Langkilde and Knight, 1998; Oh and Rudnicky, 2000; Bangalore and Rambow, 2000; Wan et al., 2009). The advantage of stochastic sentence realization over traditional rule-based realization is mainly threefold: (i) it is more robust, (ii) it usually has a significantly larger coverage; (iii) it is *per se* language- and domain-independent. Its disadvantage is that it requires at least syntactically annotated corpora of significant size (Bangalore et al., 2001). Given the aspiration of NLG to start from numeric time series or conceptual or semantic structures, syntactic annotation even does not suffice: the corpora must also be at least semantically annotated. Up to date, deep stochastic sentence realization was hampered by the lack of multiple-level annotated corpora. As a consequence, available stochastic sentence generators either take syntactic structures as input (and avoid thus the need for multiple-level annotation) (Bangalore and Rambow, 2000; Langkilde-Geary, 2002; Filippova and Strube, 2008), or draw upon hybrid models that imply a symbolic submodule which derives the syntactic representation that is then used by the stochastic submodule (Knight and Hatzivassiloglou, 1995; Langkilde and Knight, 1998).

The increasing availability of multilevel annotated corpora, such as the corpora of the shared task of the Conference on Computational Natural Language Learning (CoNLL), opens new perspectives with respect to deep stochastic sentence generation—although the fact that these corpora have not been annotated with the needs of generation in mind, may require additional adjustments, as has been, in fact, in the case of our work.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 98–106,
Beijing, August 2010

In this paper, we present a Support Vector Machine (SVM)-based multilingual dependency-oriented stochastic deep sentence realizer that uses multilingual corpora of the CoNLL '09 shared task (Hajič, 2009) for training. The sentences of these corpora are annotated with shallow semantic structures, dependency trees, and lemmata; for some of the languages involved, they also contain morphological feature annotations. The multilevel annotation allows us to take into account all levels of representation needed for linguistic generation and to model the projection between pairs of adjacent levels by separate decoders, which, in its turn, facilitates the coverage of such critical generation tasks as sentence planning, linearization, and morphologization. The presented realizer is, in principle, language-independent in that it is trainable on any multilevel annotated corpus. In this paper, we discuss its performance for Chinese, English, German, and Spanish.

The remainder of the paper is structured as follows. In Section 2, we discuss how the shallow semantic annotation in the CoNLL '09 shared task corpora should be completed in order to be suitable for generation. Section 3 presents the training setup of our realizer. Section 4 shows the individual stages of sentence realization: from the semantic structure to the syntactic structure, from the syntactic structure to the linearized structure and from the linearized structure to a chain of inflected word forms (if applicable for the language in question). Section 5 outlines the experimental set up for the evaluation of our realizer and discusses the results of this evaluation. In Section 6, finally, some conclusions with respect to the characteristics of our realizer and its place in the research landscape are drawn.

The amount of the material which comes into play makes it impossible to describe all stages in adequate detail. However, we hope that the overview provided in what follows still suffices to fully assess our proposal.

2 Completing the Semantic Annotation

The semantic annotation of sentences in CoNLL '09 shared task corpora follows the PropBank annotation guidelines (Palmer et al., 2005). Problematic from the viewpoint of generation is that this annotation is not always a connected acyclic graph. As a consequence, in these cases no valid (connected) syntactic tree can be derived. The most frequent cases of violation of the connectivity principle are not attached adjectival modifiers, determiners, adverbs, and coordinations; sometimes, the verb is not connected with its argument(s). Therefore, prior to starting the training procedure, the semantic annotation must be completed: non-connected adjectival modifiers must be annotated as predicates with their syntactic heads as arguments, determiners must be "translated" into quantifiers, detached verbal arguments must be connected with their head, etc.

Algorithm 1 displays the algorithm that completes the semantic annotations of the corpora. Each sentence x_i of the corpus I, with $i = 1, \ldots, |I|$, is annotated with its dependency tree y_i and its shallow semantic graph s_i. The algorithm traverses y_i breath-first, and examines for each node n in y_i whether n's corresponding node in s_i is connected with the node corresponding to the parent of n. If not, the algorithm connects both by a directed labeled edge. The direction and the label of the edge are selected consulting a look up table in which default labels and the orientation of the edges between different node categories are specified.

Figure 1 shows the semantic representation of a sample English sentence obtained after the application of Algorithm 1. The solid edges are the edges available in the original annotation; the dashed edges have been introduced by the algorithm. The edge labels 'A0' and 'A1' stand for "first argument" and "second argument" (of the corresponding head), respectively, 'R-A0' for "A0 realized as a relative clause", and 'AM-MNR' for "manner modifier". As can be seen, 6 out of the total of 14 edges in the complete representation of this example have been added by Algorithm 1. We still did not finish the formal evaluation of the principal changes necessary to adapt the PropBank annotation for generation, nor the quality of our completion algorithm. However, the need of an annotation with generation in mind is obvious.

Algorithm 1: Complete semantic graph

//s_i is a semantic graph and y_i a dependency tree
// $s_i = \langle N_{s_i}, L_{s_i}, E_{s_i} \rangle$, where N_{s_i} is the set of nodes
// L_{s_i} the set of edge labels
// $E_{s_i} \subseteq N_s \times N_s \times L_s$ is the set of edges
for $i \leftarrow 1$ **to** $|I|$ // iteration over the training examples
  let $r_y \in y_i$ be the root node of the dependency tree
  // initialization of the queue
  $nodeQueue \leftarrow$ *children*(r_y)
  while $nodeQueue \neq \emptyset$ **do**
    $n_y \leftarrow$ *removeFirst*(nodeQueue)
    // breath first: add nodes at the end of the queue
    nodeQueue $\leftarrow$ nodeQueue $\cup$ *children*(n_y)
    $n_{y_s} \leftarrow$ *sem*(n_y); $p_{y_s} \leftarrow$ *sem*(*parent*(n_y))
    //get the semantic equivalents of n_y and of its parent
    if not exists path(n_{y_s}, p_{y_s}) **then**
      $l \leftarrow$ *label*(n_y,*parent*(n_y))
      $l_s \leftarrow$ *look-up-sem-label*(n_{y_s}, p_{y_s}, l)
      if *look-up-sem-direction*(n_{y_s}, p_{y_s}, l_s) = "$\rightarrow$" **then**
        // add the semantic edge
        $E_s \leftarrow E_s \cup (p_{y_s}, n_{y_s}, l_s)$
      else // direction of the edge "$\leftarrow$"
        // add the semantic edge
        $E_s \leftarrow E_s \cup (n_{y_s}, p_{y_s}, l_s)$

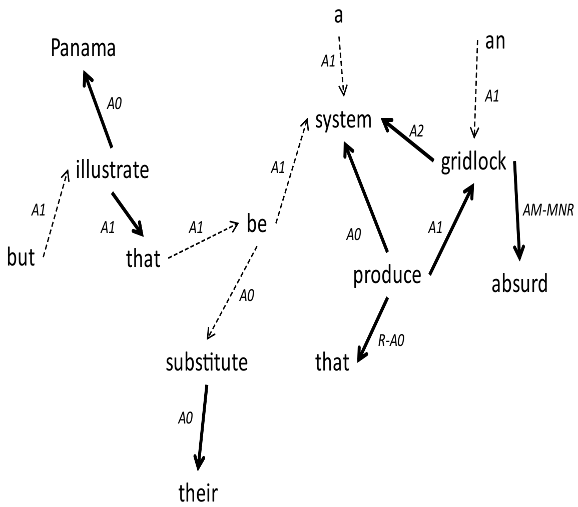


Figure 1: Semantic representation of the sentence *But Panama illustrates that their substitute is a system that produces an absurd gridlock.* after completion

3 Realizer Training Setup

Figure 2 shows the training setup of our realizer. For each level of annotation, an SVM feature extractor and for each pair of adjacent levels of annotation, an SVM decoder is defined. The Sem-Synt decoder constructs from a semantic graph the corresponding dependency tree. The Synt-Linearization decoder derives from a dependency tree a chain of lemmata, i.e., determines the word order within the sentence. The Linearization-Morph decoder generates the inflected word form for each lemma in the chain. Both the feature extractors and the decoders are language-independent, which makes the realizer applicable to any language for which multilevel-annotated corpora are available.

To compute the score of the alternative realizations by each decoder, we apply MIRA (Margin Infused Relaxed Algorithm) to the features provided by the feature extractors. MIRA is one of the most successful large-margin training techniques for structured data (Crammer et al., 2006). It has been used, e.g., for dependency parsing, semantic role labelling, chunking and tagging. Since we have similar feature sets (of comparable size) as those for which MIRA has proven to work well, we assume that it will also perform well for sentence realization. Unfortunately, due to the lack of space, we cannot present here the instantiation of MIRA for all stages of our model. For illustration, Algorithm 2 outlines it for morphological realization.

The morphologic realization uses the minimal string edit distance (Levenshtein, 1966) to map lemmata to word forms. As input to the MIRA-classifier, we use the lemmata of a sentence, its dependency tree and the already ordered sentence. The characters of the input strings are reversed since most of the changes occur at the end of the words and the string edit scripts work relatively to the beginning of the string. For example, to calculate the minimal string edit distance between the lemma *go* and the form *goes*, both are first reversed by the function **compute-edit-dist** and then the minimal string edit script between *og* and *seog* is computed. The resulting script is *Ie0Is0*. It translates into the operations 'insert *e* at the position 0 of the input string' and 'insert *s* at the position 0'.

Before MIRA starts, we compute all minimal edit distance scripts to be used as classes of MIRA. Only scripts that occur more often than twice are used. The number of the resulting edit scripts is language-dependent; e.g., we get about

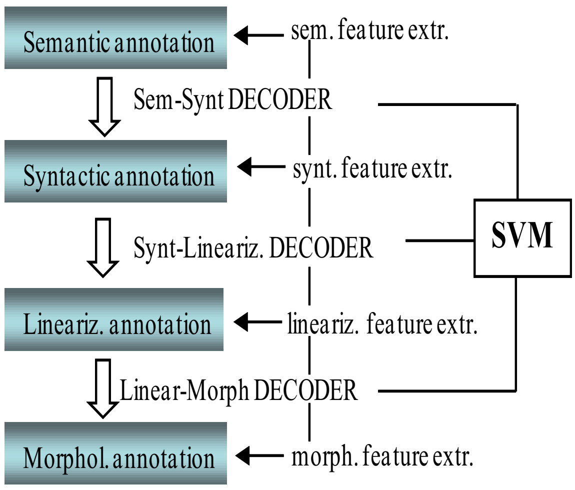


Figure 2: Realizer training scenario setup

1500 scripts for English and 2500 for German.

The training algorithms typically perform 6 iterations (*epochs*) over the training examples. For each training example, a minimal edit script is selected. If this script is different from the gold script, the features of the gold script are calculated and the weight vector of the SVM is adjusted according to the difference between the predicted vector and the *gold feature vector*. The classification task consists then in finding the classification script that maps the lemma to the correct word form. For this purpose, the classifier scores each of the minimal edit scripts according to the input, choosing the one with the highest score.

4 Sentence Generation

Sentence generation that starts from a given semantic structure as input consists in the application of the previously trained SVM decoders in sequence in order to realize the following sequence of mappings:

SemStr $\rightarrow$ *SyntStr* $\rightarrow$ *LinearStr* $\rightarrow$ *Surface*

4.1 Semantic Generation

Algorithm 3 shows the algorithm for semantic generation, i.e., the derivation of a dependency tree from a semantic structure. It is a beam search that creates a maximum spanning tree. In the first step, a seed tree consisting of one edge is built. In each of the subsequent steps, this tree is extended by one node. For the decision, which node

Algorithm 2: Morphological realization training with MIRA

// y_i, l_i; y_i is a dependency tree, l_i lemmatized sentence
script-list $\leftarrow$ {} //initialize the script-list
for $i \leftarrow 1$ **to** $|I|$ // iteration over the training examples
  for $l \leftarrow 1$ to $|l_i|$ **do**//// iteration over the lemmata of l_i
    lemma$_l$ $\leftarrow$ *lower-case* (l_i, l)
    //ensure that all lemmata start with a lower case letter
    script $\leftarrow$ *compute-edit-dist-script*(lemma$_l$, *form*(l_i, l))
    if script $\notin$ script-list
      script-list $\leftarrow$ script-list $\cup$ { script }
for $k \leftarrow 1$ **to** E // E = number of traininig epochs
  for $i \leftarrow 1$ **to** $|I|$ // iteration over the training examples
    for $l \leftarrow 1$ to $|l_i|$ **do**
      script$_p$ $\leftarrow$ *predict-script*(l_i, y_i, l)
      script$_g$ $\leftarrow$ *edit-dist-script*(lemma$_l$, form(l_i, l))
      if script$_p$ $\neq$ script$_g$ **then**
      // update the weight vector v and the vector w, which
      // averages over all collected weight vectors acc.
      // to diff. of the predicted and gold feature vector
      update w, v according to $\Delta(\phi(\text{script}_p), \phi(\text{script}_g))$
      //with $\phi(\text{script}_p)$, $\phi(\text{script}_g)$ as feature vectors of
      //script$_p$ and script$_g$, respectively

is to be attached next and to which node, we consider the highest scoring options. This procedure works well since nodes that are close in the semantic structure are usually close in the syntactic tree as well. Therefore subtrees that contain those nodes are considered first.

Unlike the traditional n-gram based stochastic realizers such as (Langkilde and Knight, 1998), we use for the score calculation structured features composed of the following elements: (i) the lemmata, (ii) the **dist**ance between the starting node s and the target node t, (iii) the **dir**ection of the path (if the path has a direction), (iv) the sorted **bag** of in-going edges labels without repitition, (v) the **path** of edge labels between source and target node.

The composed structured features are:

- label+dist(s, t)+dir
- label+dist(s, t)+lemma$_s$+dir
- label+dist(s, t)+lemma$_t$+dir
- label+dist(s, t)+lemma$_s$+lemma$_t$+dir
- label+dist(s, t)+bag$_s$+dir
- label+dist(s, t)+bag$_t$+dir
- label+path(s, t)+dir

#	word-pairs(w_1,w_2)	#	n-grams
1	label_{w1}+label_{w2}	13	PoS_1+PoS_2+PoS_3
2	label_{w1}+lemma_1	14	PoS_1+PoS_2+PoS_3+dist
3	label_{w1}+lemma_2	15	lemma_1+lemma_2+lemma_3
4	label_{w2}+lemma_1	16	lemma_1+lemma_2+lemma_3+dist
5	label_{w2}+lemma_2	17	lemma_1+lemma_3+head(w1,w2,w3)
6	PoS_1+PoS_2	18	lemma_1+lemma_3+head(w1,w2,w3)+dist
7	PoS_1+PoS_2+head(w_1,w_2)	19	label_1+label_2+label_3+head(w1,w2,w3)
8	label_{w1}+label_{w2}+PoS_1+head(w_1,w_2)	20	label_1+label_2+label_3+head(w1,w2,w3)+dist
9	label_{w1}+label_{w2}+PoS_2+head(w_1,w_2)	21	label_1+label_2+label_3+lemma_1+PoS_2+head(w1,w2,w3)
10	label_{w1}+label_{w2}+PoS_1+PoS_2+head(w_1,w_2)	22	label_1+label_2+label_3+lemma_1+PoS_2+head(w1,w2,w3)+dist
11	label_{w1}+label_{w2}+PoS_1+$\#\text{children}_2$+head(w_1,w_2)	23	label_1+label_2+label_3+lemma_2+PoS_1+head(w1,w2,w3)
12	label_{w1}+label_{w2}+PoS_2+$\#\text{children}_1$+head(w_1,w_2)	24	label_1+label_2+label_3+lemma_2+PoS_1+head(w1,w2,w3)+dist

#	global features for constituents
25	**if** |constituent| > 1 **then** label_{1st}+label_{last}+label_{last-1}+PoS_{first}+PoS_{last}+PoS_{head}
26	**if** |constituent| > 2 **then** label_{1st}+label_{2d}+label_{3d}+PoS_{last}+PoS_{last-1}+PoS_{head}+contains-?
27	**if** |constituent| > 2 **then** label_{1st}+label_{2d}+label_{3d}+PoS_{last}+PoS_{last-1}+lemma_{head}+contains-?
28	**if** |constituent| > 3 **then** PoS_{1st}+PoS_{2d}+PoS_{3d}+PoS_{4th}+PoS_{last}+label_{head}+contains-?+pos-head
29	**if** |constituent| > 3 **then** PoS_{last}+PoS_{last-1}+PoS_{last-2}+PoS_{last-3}+PoS_{first}+label_{head}+contains-?+pos-head
30	PoS_{first}+PoS_{last}+lemma_{first}+lemma_{last}+lemma_{head}+contains-?+pos-head

Table 1: Feature schemas used for linearization ($label_w$ is the label of the in-going edge to a word w in the dependency tree; $lemma_w$ is the lemma of w, and PoS_w is the part-of-speech tag of w; *head*(w_1,w_2, ...) is a function which is 1 if w_1 is the head, 2 if w_2 is the head, etc. and else 0; *dist* is the position within the constituent; *contains-?* is a boolean value which is true if the sentence contains a question mark and false otherwise; *pos-head* is the position of the head in the constituent)

4.2 Dependency Tree Linearization

Since we use unordered dependency trees as syntactic structures, our realizer has to find the optimal linear order for the lexemes of each dependency tree. Algorithm 4 shows our linearization algorithm. To order the dependency tree, we use a one classifier-approach for all languages—in contrast to, e.g., Filippova and Strube (2009), who use a two-classifier approach for German.[1]

The algorithm is again a beam search. It starts with an elementary list for each node of the dependency tree. Each elementary list is first extended by the children of the node in the list; then, the lists are extended stepwise by the children of the newly added nodes. If the number of lists during this procedure exceeds the threshold of 1000, the lists are sorted in accordance with their score, and the first 1000 are kept. The remaining lists are removed. Afterwards, the score of each list is adjusted according to a global score function which takes into account complex features such as the first word of a consitutent, last word, the head, and the edge label to the head (cf. Table 1 for the list of the features). Finally, the nodes of the dependency tree are ordered with respect to the highest ranked lists.

Only in a very rare case, the threshold of the beam search is exceeded. Even with a rich feature set, the procedure is very fast. The linearization takes about 3 milliseconds in average per dependency tree on a computer with a 2.8 Ghz CPU.

4.3 Morphological Realization

The morphological realization algorithm selects the edit script in accordance with the highest score for each lemma of a sentence obtained during training (see Algorithm 2 above) and applies then the scripts to obtain the word forms; cf. Algorithm 5.

Table 2 lists the feature schemas used for morphological realization.

5 Experiments

To evaluate the performance of our realizer, we carried out experiments on deep generation of Chinese, English, German and Spanish, starting from CoNLL ’09 shared task corpora. The size of the test sets is listed in Table 3.[2]

[1] We decided to test at this stage of our work a uniform technology for all languages, even if the idiosyncrasies of some languages may be handled better by specific solutions.

[2] As in (Langkilde-Geary, 2002) and (Ringger et al., 2004), we used Section 23 of the WSJ corpus as test set for English.

Algorithm 3: Semantic generation

```
//s_i, y semantic graph and its dependency tree
for i ← 1 to |I| // iteration over the training examples
  // build an initial tree
  for all n_1 ∈ s_i do
  trees ← {} // initialize the constructed trees list
    for all n_2 ∈ s_i do
      if n_1 ≠ n_2 then
        for all l ∈ dependency-labels do
          trees = trees ∪ {(synt(n_1),synt(n_2),l)}
  trees ← sort-trees-descending-to-score(trees)
  trees ← look-forward(1000,sublist(trees,20))
  //assess at most 1000 edges of the 20 best trees
  tree ← get-best-tree-due-to-score(trees)
  (s,t,l) ← first-added-edge(tree)
  // create the best tree
  best-tree ← (s,t,l)
  // compute the nodes that still need to be attached
  rest ← nodes(s_i) - {s, t}
  while rest ≠ ∅ do
    trees ← look-forward(1000,best-tree,rest)
    tree ← get-best-tree-due-to-score(trees)
    (s,t,l) ← first-added-edge(tree)
    best-tree ← best-tree ∪ { (s,t,l) }
    if (root(s,best-tree)) then rest ← rest - {s}
    else rest ← rest - {t}
```

The performance of both the isolated stages and the realizer as a whole has been assessed.

5.1 Evaluation Metrics

In order to measure the correctness of the semantics to syntax mapping, we use the unlabeled and labeled attachment score as it commonly used in dependency parsing. The labeled attachment score (LAS) is the proportion of tokens that are assigned both the correct head and the correct edge label. The unlabeled attachment score (ULA) is the proportion of correct tokens that are assigned the correct head.

To assess the quality of linearization, we use three different evaluation metrics. The first metric is the per-phrase/per-clause accuracy (*acc snt.*), which facilitates the automatic evaluation of results:

$$acc = \frac{correct\ constituents}{all\ constituents}$$

As second evaluation metric, we use a metric related to the edit distance:

$$di = 1 - \frac{m}{total\ number\ of\ words}$$

(with m as the minimum number of deletions combined with insertions to obtain the correct order (Ringger et al., 2004)).

Algorithm 4: Dependency tree linearization

```
//y_i a dependency tree
for i ← 1 to |I| // iteration over the training examples
  // iterate over all nodes of the dependency tree y_i
  for n ← 1 to |y_i| do
    subtree_n ← children(n) ∪ {n}
    ordered-lists_n ← {} // initialize
    for all m ∈ subtree_n do
      beam ← {}
      for all l ∈ ordered-lists do
        beam ← beam  ∪ { append(clone(l),m)}
      for all l ∈ ordered-lists do
        score(l) ← compute-score-for-word-list(l)
      sort-lists-descending-to-score(beam,score)
      if | beam | > beam-size then
        beam ← sublist(0,1000,beam)
      ordered-lists_n ← beam
    score_g(l) ← score(l) + compute-global-score(l)
    sort-lists-descending-in-score(beam,score_g)
```

Algorithm 5: Morphological realization

```
// y_i a dependency tree, and l_i an ordered list of lemmata
for l ← 1 to |l_i| do
  script_p ← predict-script(l_i,y_i,l)
  form_l ← apply-edit-dist-script(lemma_l, script_p)
```

To be able to compare our results with (He et al., 2009) and (Ringger et al., 2004), we use the BLEU score as a third metric.

For the asessment of the quality of the word form generation, we use the accuracy score. The accuracy is the ratio between correctly generated word forms and the entire set of generated word forms.

For the evaluation of the sentence realizer as a whole, we use the BLEU metric.

5.2 Experimental Results

Table 4 displays the results obtained for the isolated stages of sentence realization and of the realization as a whole, with reference to a baseline and to some state-of-the-art works. The baseline is the deep sentence realization over all stages starting from the original semantic annotation in the CoNLL '09 shared task corpora.

Note, that our results are not fully comparable with (He et al., 2009; Filippova and Strube, 2009) and (Ringger et al., 2004), respectively, since the data are different. Furthermore, Filippova and Strube (2009) linearize only English sentences

#	features
1	es+lemma
2	es+lemma+m.feats
3	es+lemma+m.feats+POS
4	es+lemma+m.feats+POS+position
5	es+lemma+(lemma+1)+m.feats
6	es+lemma+(lemma+1)+POS
7	es+lemma+(m.feats-1)+(POS-1)
8	es+lemma+(m.feats-1)+(POS-1)+position
9	es+m.feats+(m.feats-1)
10	es+m.feats+(m.feats+1)
11	es+lemma+(m.feats-1)
12	es+m.feats+(m.feats-1)+(m.feats-2)
13	es+m.feats+POS
14	es+m.feats+(m.feats+1)
15	es+m.feats+(m.feats+1)+lemma
16	es+m.feats
17	es+e0+e1+m.feats
18	es+e0+e1+e2+m.feats
19	es+e0+e1+e2+e3+m.feats
20	es+e0+e1+e2+e3+e4+m.feats
21	es+e0+m.feats

Table 2: Feature schemas used for morphological realization

Chinese	English	German	Spanish
2556	2400	2000	1725

Table 3: The number of sentences in the test sets used in the experiments

that do not contain phrases that exceed 20,000 linearization options—which means that they filter out about 1% of the phrases.

For Spanish, to the best of our knowledge, no linearization experiments have been carried out so far. Therefore, we cannot contrast our results with any reference work.

As far as morphologization is concerned, the performance achieved by our realizer for English is somewhat lower than in (Minnen et al., 2001) (97.8% vs. 99.8% of accuracy). Note, however, that Minnen et al. describe a combined analyzer-generator, in which the generator is directly derived from the analyzer, which makes both approaches not directly comparable.

5.3 Discussion

The overall performance of our SVM-based deep sentence generator ranges between 0.611 (for German) and 0.688 (for Chinese) of the BLEU score. HALogen's (Langkilde-Geary, 2002) scores range between 0.514 and 0.924, depending on the completeness of the input. The figures are not directly comparable since HALogen takes as input syntactic structures. However, it gives us an idea where our generator is situated.

Traditional linearization approaches are rule-based; cf., e.g., (Bröker, 1998; Gerdes and Kahane, 2001; Duchier and Debusmann, 2001), and (Bohnet, 2004). More recently, statistic language models have been used to derive word order, cf. (Ringger et al., 2004; Wan et al., 2009) and (Filippova and Strube, 2009). Because of its partially free order, which is more difficult to handle than fixed word order, German has often been worked with in the context of linearization. Filippova and Strube (2009) adapted their linearization model originally developed for German to English. They use two classifiers to determine the word order in a sentence. The first classifier uses a trigram LM to order words within constituents, and the second (which is a maximum entropy classifier) determines the order of constituents that depend on a finite verb. For English, we achieve with our SVM-based classifier a better performance. As mentioned above, for German, Filippova and Strube (2009)'s two classifier approach pays off because it allows them to handle non-projective structures for the *Vorfeld* within the field model. It is certainly appropriate to optimize the performance of the realizer for the languages covered in a specific application. However, our goal has been so far different: to offer an off-the-shelf language-independent solution.

The linearization error analysis, first of all of German and Spanish, reveals that the annotation of coordinations in corpora of these languages as 'X $\leftarrow$ *and/or/*... $\rightarrow$ Y' is a source of errors. The "linear" annotation used in the PropBank ('X $\rightarrow$ *and/or/*... $\rightarrow$ Y') appears to facilitate higher quality linearization. A preprocessing stage for automatic conversion of the annotation of coordinations in the corpora would have certainly contributed to a higher quality. We refrained from doing this because we did not want to distort the figures.

The morphologization error analysis indicates a number of error sources that we will address in the process of the improvement of the model. Among those sources are: quotes at the beginning of a sentence, acronyms, specific cases of starting capital letters of proper nouns (for English and Spanish), etc.

	Chinese	English	German	Spanish
Semantics-Syntax (ULA/LAS)	95.71/86.29	94.77/89.76	95.46/82.99	98.39/93.00
Syntax-Topology (di/acc)	0.88/64.74	0.91/74.96	0.82/50.5	0.83/52.77
Syntax-Topology (BLEU)	0.85	0.894	0.735	0.78
Topology-Morphology (accuracy=correct words/all words)	–	97.8	97.49	98.48
All stages (BLEU)	0.688	0.659	0.611	0.68
Baseline (BLEU)	0.12	0.18	0.11	0.14
Syntax-Topology (He et al., 2009) (di/acc)	0.89/–	–	–	–
Syntax-Topology (He et al., 2009) (BLEU)	0.887	–	–	–
Syntax-Topology (Filippova and Strube, 2009) (di/acc)	–	0.88/67	0.87/61	–
Syntax-Topology (Ringger et al., 2004) (BLEU)	–	0.836	–	–

Table 4: Quality figures for the isolated stages of deep sentence realization and the complete process.

As far as the contrastive evaluation of the quality of our morphologization stage is concerned, it is hampered by the fact that for the traditional manually crafted morphological generators, it is difficult to find thorough quantitative evaluations, and stochastic morphological generators are rare.

As already repeatedly pointed out above, so far we intentionally refrained from optimizing the individual realization stages for specific languages. Therefore, there is still quite a lot of room for improvement of our realizer when one concentrates on a selected set of languages.

6 Conclusions

We presented an SVM-based stochastic deep multilingual sentence generator that is inspired by the state-of-the-art research in semantic parsing. It uses similar techniques and relies on the same resources. This shows that there is a potential for stochastic sentence realization to catch up with the level of progress recently achieved in parsing technologies.

The generator exploits recently available multilevel-annotated corpora for training. While the availability of such corpora is a condition for deep sentence realization that starts, as is usually the case, from semantic (predicate-argument) structures, we discovered that current annotation schemata do not always favor generation such that additional preprocessing is necessary. This is not surprising since stochastic generation is a very young field. An initiative of the generation community would be appropriate to influence future multilevel annotation campaigns or to feed back the enriched annotations to the "official" resources.[3]

The most prominent features of our generator are that it is *per se* multilingual, it achieves an extremely broad coverage, and it starts from abstract semantic structures. The last feature allows us to cover a number of critical generation issues: sentence planning, linearization and morphological generation. The separation of the semantic, syntactic, linearization and morphological levels of annotation and their modular processing by separate SVM decoders also facilitates a subsequent integration of other generation tasks such as referring expression generation, ellipsis generation, and aggregation. As a matter of fact, this generator instantiates the Reference Architecture for Generation Systems (Mellish et al., 2006) for linguistic generation.

A more practical advantage of the presented deep stochastic sentence generator (as, in principle, of all stochastic generators) is that, if trained on a representative corpus, it is domain-independent. As rightly pointed out by Belz (2008), traditional wide coverage realizers such as KPML (Bateman et al., 2005), FUF/SURGE (Elhadad and Robin, 1996) and RealPro (Lavoie and Rambow, 1997), which were also intended as off-the-shelf plug-in realizers still tend to require a considerable amount of work for integration and fine-tuning of the grammatical and lexical resources. Deep stochastic sentence realizers have the potential to become real off-the-shelf modules. Our realizer is freely available for download at `http://www.recerca.upf.edu/taln`.

[3] We are currently working on a generation-oriented multilevel annotation of corpora for a number of languages. The corpora will be made available to the community.

Acknowledgments

Many thanks to the three anonymous reviewers for their very valuable comments and suggestions.

References

Bangalore, S. and O. Rambow. 2000. Exploiting a Probabilistic Hierarchical Model for Generation. In *Proceedings of COLING '00*, pages 42–48.

Bangalore, S., J. Chen, and O. Rambow. 2001. Impact of Quality and Quantity of Corpora on Stochastic Generation. In *Proceedings of the EMNLP Conference*, pages 159–166.

Bateman, J.A., I. Kruijff-Korbayová, and G.-J. Kruijff. 2005. Multilingual Resource Sharing Across Both Related and Unrelated Languages: An Implemented, Open-Source Framework for Practical Natural Language Generation. *Research on Language and Computation*, 15:1–29.

Belz, A. 2008. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*, 14(4):431–455.

Bohnet, B. 2004. A graph grammar approach to map between dependency trees and topological models. In *Proceedings of the IJCNLP*, pages 636–645.

Bröker, N. 1998. Separating Surface Order and Syntactic Relations in a Dependency Grammar. In *Proceedings of the COLING/ACL '98*.

Crammer, K., O. Dekel, S. Shalev-Shwartz, and Y. Singer. 2006. Online Passive-Aggressive Algorithms. *Journal of Machine Learning Research*, 7:551–585.

Duchier, D. and R. Debusmann. 2001. Topological dependency trees: A constraint-based account of linear precedence. In *Proceedings of the ACL*.

Elhadad, M. and J. Robin. 1996. An overview of SURGE: A reusable comprehensive syntactic realization component. Technical Report TR 96-03, Department of Mathematics and Computer Science, Ben Gurion University.

Filippova, K. and M. Strube. 2008. Sentence fusion via dependency graph compression. In *Proceedings of the EMNLP Conference*.

Filippova, K. and M. Strube. 2009. Tree linearization in English: Improving language model based approaches. In *Proceedings of the NAACL '09 and HLT, Short Papers*, pages 225–228.

Gerdes, K. and S. Kahane. 2001. Word order in German: A formal dependency grammar using a topological hierarchy. In *Proceedings of the ACL*.

Hajič, J. et al. 2009. The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages. In *Proceedings of the CoNLL*.

He, W., H. Wang, Y. Guo, and T. Liu. 2009. Dependency based chinese sentence realization. In *Proceedings of the ACL and of the IJCNLP of the AFNLP*, pages 809–816.

Knight, K. and V. Hatzivassiloglou. 1995. Two-level, many paths generation. In *Proceedings of the ACL*.

Langkilde, I. and K. Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the COLING/ACL*, pages 704–710.

Langkilde-Geary, I. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the Second INLG Conference*, pages 17–28.

Lavoie, B. and O. Rambow. 1997. A fast and portable realizer for text generation systems. In *Proceedings of the 5th Conference on ANLP*.

Levenshtein, V.I. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics*, 10:707–710.

Mellish, C., D. Scott, L. Cahill, D. Paiva, R. Evans, and M. Reape. 2006. A reference architecture for natural language generation systems. *Natural Language Engineering*, 12(1):1–34.

Minnen, G., J. Carroll, and D. Pearce. 2001. Applied morphological processing for English. *Natural Language Engineering*, 7(3):207–223.

Oh, A.H. and A.I. Rudnicky. 2000. Stochastic language generation for spoken dialogue systems. In *Proceedings of the ANL/NAACL Workshop on Conversational Systems*, pages 27–32.

Palmer, M., D. Gildea, and P. Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105.

Ringger, E., M. Gamon, R.C. Moore, D. Rojas, M. Smets, and S. Corston-Oliver. 2004. Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of COLING*, pages 673–679.

Wan, S., M. Dras, Dale R., and C. Paris. 2009. Improving Grammaticality in Statistical Sentence Generation: Introducing a Dependency Spanning Tree Algorithm with an Argument Satisfaction Model. In *Proceedings of the EACL '09*, pages 852–860.

Towards an optimal weighting of context words based on distance

Bernard Brosseau-Villeneuve*#, Jian-Yun Nie*, Noriko Kando#
* Université de Montréal, Email: {brosseab, nie}@iro.umontreal.ca
# National Institute of Informatics, Email: {bbrosseau, kando}@nii.ac.jp

Abstract

Word Sense Disambiguation (WSD) often relies on a context model or vector constructed from the words that co-occur with the target word within the same text windows. In most cases, a fixed-sized window is used, which is determined by trial and error. In addition, words within the same window are weighted uniformly regardless to their distance to the target word. Intuitively, it seems more reasonable to assign a stronger weight to context words closer to the target word. However, it is difficult to manually define the optimal weighting function based on distance. In this paper, we propose a unsupervised method for determining the optimal weights for context words according to their distance. The general idea is that the optimal weights should maximize the similarity of two context models of the target word generated from two random samples. This principle is applied to both English and Japanese. The context models using the resulting weights are used in WSD tasks on Semeval data. Our experimental results showed that substantial improvements in WSD accuracy can be obtained using the automatically defined weighting schema.

1 Introduction

The meaning of a word can be defined by the words that accompany it in the text. This is the principle often used in previous studies on Word Sense Disambiguation (WSD) (Ide and Véronis, 1998; Navigli, 2009). In general, the accompanying words form a context vector of the target word, or a probability distribution of the context words. For example, under the unigram bag-of-words assumption, this means building $p(x|t) = \frac{count(x,t)}{\sum_{x'} count(x',t)}$, where $count(x,t)$ is the count of co-occurrences of word x with the target word t under a certain criterion. In most studies, x and t should co-occur within a window of up to k words or sentences. The bounds are usually selected in an ad-hoc fashion to maximize system performance. Occurrences inside the window often weight the same without regard to their position. This is counterintuitive. Indeed, a word closer to the target word generally has a greater semantic constraint on the target word than a more distant word. It is however difficult to define the optimal weighting function manually. To get around this, some systems add positional features for very close words. In information retrieval, to model the strength of word relations, some studies have proposed non-uniform weighting methods of context words, which decrease the importance of more distant words in the context vector. However, the weighting functions are defined manually. It is unclear that these functions can best capture the impact of the context words on the target word.

In this paper, we propose an unsupervised method to automatically learn the optimal weight of a word according to its distance to the target word. The general principle used to determine such weight is that, if we randomly determine two sets of windows containing the target word from the same corpus, the meaning – or mixture of meanings for polysemic words – of the target word in the two sets should be similar. As the context model – a probability distribution for the context words – determines the meaning of the target word, the context models generated from the two sets should also be similar. The weights of context words at different distance are therefore de-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 107–115,
Beijing, August 2010

termined so as to maximize the similarity of context models generated from the two sets of samples. In this paper, we propose a gradient descent method to find the optimal weights. We will see that the optimal weighting functions are different from those used in previous studies. Experimentation on Semeval-2007 English and Semeval-2010 Japanese lexical sample task data shows that improvements can be attained using the resulting weighting functions on simple Naïve Bayes (NB) systems in comparison to manually selected functions. This result validates the general principle we propose in this paper.

The remainder of this paper is organized as follows: typical uses of text windows and related work are presented in Section 2. Our method is presented in Section 3. In Section 4 to 6, we show experimental results on English and Japanese WSD. We conclude in Section 7 with discussion and further possible extensions.

2 Uses of text windows

Modeling the distribution of words around one target word, which we call context model, has many uses. For instance, one can use it to define a co-occurrence-based stemmer (Xu and Croft, 1998), which uses window co-occurrence statistics to calculate the best equivalence classes for a group of word forms. In the study of Xu and Croft, they suggest using windows of up to 100 words. Context models are also widely used in WSD. For example, top performing systems on English WSD tasks in Semeval-2007, such as NUS-ML (Cai et al., 2007), all made use of bag-of-words features around the target word. In this case, they found that the best results can be achieved using a window size of 3.

Both systems limit the size of their windows for different purposes. The former uses a large size in order to model the topic of the documents containing the word rather than the word's meaning. The latter would limit the size because bag-of-words features further from the target word would not be sufficiently related to its meaning (Ide and Véronis, 1998). We see that there is a compromise between taking fewer, highly related words, or taking more, lower quality words. However, there is no principled way to determine the optimal size of windows. The size is determined by trial and error.

A more questionable aspect in the above systems is that for bag-of-words features, all words in a window are given equal weights. This is counterintuitive. One can easily understand that a context word closer to the target word *generally* imposes a stronger constraint on the meaning of the latter, than a more distant context word. It is then reasonable to define a weighting function that decreases along with distance. Several studies in information retrieval (IR) have proposed such functions to model the strength of dependency between words. For instance, Gao et al. (2002) proposed an exponential decay function to capture the strength of dependency between words. This function turns out to work better than the uniform weighting in the IR experiments.

Song and Bruza (2003) used a fixed-size sliding window to determine word co-occurrences. This is equivalent to define a linear decay function for context words. The context vectors defined this way are used to estimate similarity between words. A use of the resulting similarity in query expansion in IR turned out to be successful (Bai et al., 2005).

In a more recent study, Lv and Zhai (2009) evaluated several kernel functions to determine the weights of context words according to distance, including Gaussian kernel, cosine kernel, and so on. As for the exponential and linear decaying functions, all these kernel functions have fixed shapes, which are determined manually.

Notice that the above functions have only been tested in IR experiments. It is not clear how these functions perform in WSD. More importantly, all the previous studies have investigated only a limited number of weighting functions for context words. Although some improvements using these functions have been observed in IR, it is not clear whether the functions can best capture the true impact of the context words on the meaning of the target word. Although the proposed functions comply with the general principle that closer words are more important than more distant words, no principled way has been proposed to determine the particular shape of the function for different languages and collections.

In this paper, we argue that there is indeed a hidden weighting function that best capture the impact of context words, but the function cannot be defined manually. Rather, the best function should be the one that emerges naturally from the data. Therefore, we propose an unsupervised method to discover such a function based on the following principle: the context models for a target word generated from two random samples should be similar. In the next section, we will define in detail how this principle is used.

3 Computing weights for distances

In this section, we present our method for choosing how much a word occurrence should count in the context model according to its distance to the target word. In this study, for simplicity, we assume that all word occurrences at a given distance count equally in the context model. That is, we ignore other features such as POS-tags, which are used in other studies on WSD.

Let $\mathcal{C}$ be a corpus, W a set of text windows for the target word w, $c_{W,i,x}$ the count of occurrences of word x at distance i in W, $c_{W,i}$ the sum of these counts, and α_i the weight put on one word occurrence at distance i. Then,

$$P_{ML,W}(x) = \frac{\sum_i \alpha_i c_{W,i,x}}{\sum_i \alpha_i c_{W,i}} \tag{1}$$

is the maximum likelihood estimator for x in the context model of w. To counter the zero probability problem, we apply Dirichlet smoothing with the collection language model as a prior:

$$P_{Dir,W}(x) = \frac{\sum_i \alpha_i c_{W,i,x} + \mu_W P(x|\mathcal{C})}{\sum_i \alpha_i c_{W,i} + \mu_W} \tag{2}$$

The pseudo-count μ_W can be a constant, or can be found by using Newton's method, maximizing the log likelihood via leave-one-out estimation:

$$\mathcal{L}_{-1}(\mu|W,\mathcal{C}) = \sum_i \sum_{x\in V} \alpha_i c_{W,i,x} \log \frac{\alpha_i c_{W,i,x} - \alpha_i + \mu P(x|\mathcal{C})}{\sum_j \alpha_j c_{W,j} - \alpha_i + \mu}$$

The general process, which we call automatic Dirichlet smoothing, is similar to that described in (Zhai and Lafferty, 2002).

To find the best weights for our model we propose the following process:

- Let T be the set of all windows containing the target word. We randomly split this set into two sets A and B.
- We want to find $\alpha^\star$ that maximizes the similarity of the models obtained from the two sets, by minimizing their mutual cross entropy:

$$\begin{aligned} l(\alpha) &= H(P_{ML,A}, P_{Dir,B}) + \\ &\quad H(P_{ML,B}, P_{Dir,A}) \end{aligned} \tag{3}$$

In other words, we want α_i to represent how much an occurrence at distance i models the context better than the collection language model, whose counts are weighted by the Dirichlet parameter. We hypothesize that target words occur in limited contexts, and as we get farther from them, the possibilities become greater, resulting in sparse and less related counts. Since two different sets of the same word are essentially noisy samples of the same distribution, the weights maximizing their mutual generation probabilities should model this phenomenon.

One may wonder why we do not use a distribution similarity metric such as Kullback–Leibler (KL) divergence or Information Radius (IRad). The reason is that with enough word occurrences (big windows or enough samples), the most similar distributions are found with uniform weights, when all word counts are used. KL divergence is especially problematic as, since it requires smoothing, the weights will converge to the degenerate weights $\alpha = 0$, where only the identical smoothing counts remain. Entropy minimization is therefore needed in the objective function.

To determine the optimal weight of α_i, we propose a simple gradient descent minimizing (3) over α. The following are the necessary derivatives:

$$\begin{aligned} \frac{\partial l}{\partial \alpha_i} &= \frac{\partial H(P_{ML,A}, P_{Dir,B})}{\partial \alpha_i} + \\ &\quad \frac{\partial H(P_{ML,B}, P_{Dir,A})}{\partial \alpha_i} \end{aligned}$$

$$\frac{\partial H\left(P_{ML,W}, P_{Dir,(T-W)}\right)}{\partial \alpha_i} =$$

$$- \sum_{x \in V} \left[\frac{\partial P_{ML,W}(x)}{\partial \alpha_i} \log P_{Dir,(T-W)}(x) + \frac{\partial P_{Dir,(T-W)}(x)}{\partial \alpha_i} \times \frac{P_{ML,W}(x)}{P_{Dir,(T-W)}(x)} \right]$$

$$\frac{\partial P_{ML,W}(x)}{\partial \alpha_i} = \frac{c_{W,i,x} - P_{ML,W}(x) c_{W,i}}{\sum_j \alpha_j c_{W,j}}$$

$$\frac{\partial P_{Dir,W}(x)}{\partial \alpha_i} = \frac{c_{W,i,x} - P_{Dir,W}(x) c_{W,i}}{\sum_j \alpha_j c_{W,j} + \mu_W}$$

We use stochastic gradient descent: one word is selected randomly, it's gradient is computed, a small gradient step is done and the process is repeated. A pseudo-code of the process can be found in Algorithm 1.

Algorithm 1 LearnWeight($\mathcal{C}, \eta, \epsilon$)

```
α ← 1^k
repeat
    T ←{Get windows for next word}
    (A, B) ←RandomPartition(T)
    for W in A, B do
        P_{ML,W} ←MakeML(W,α)
        μ_W ←ComputePseudoCount(W,C)
        P_{Dir,W} ←MakeDir( P_{ML,W}, μ_W, C)
    end for
    grad ← ∇H(P_{ML,A}, P_{Dir,B}) + ∇H(P_{ML,B}, P_{Dir,A})
    α ← α − η grad/||grad||
until ∃α_i < ε
return α/max{α_i}
```

Now, as the objective function would eventually go towards putting nearly all weight on α_1, we hypothesize that the farthest distances should have a near-zero contribution, and determine the stop criterion as having one weight go under a small threshold. Alternatively, a control set of held out words can be used to observe the progress of the objective function or the gradient length. When more and more weight is put on the few closest positions, the objective function and gradient depends on less counts and will become less stable. This can be used as a stop criterion.

The above weight learning process is applied on an English collection and a Japanese collection with $\eta = \epsilon = 0.001$, and $\mu = 1000$. In the next sections, we will describe both resulting weighting functions in the context of WSD experiments.

4 Classifiers for supervised WSD tasks

Since we use the same systems for both English and Japanese experiments, we will briefly discuss the used classifiers in this section. In both tasks, the objective is to maximize WSD accuracy on held-out data, given that we have a set of training text passages containing a sense-annotated target word.

The first of our baselines, the *Most Frequent Sense* (MFS) system always selects the most frequent sense in the training set. It gives us a lower bound on system accuracies.

Naïve Bayes (NB) classifiers score classes using the Bayes formula under a feature independence assumption. Let w be the target word in a given window sample to be classified, the scoring formula for sense class S is:

$$Score(w, S) = P(S) P_{Tar}(w|S)^{\lambda_{Tar}} \times \prod_{x_i \in context(w)} P_{Con}(x_i|S)^{\lambda_{Con} \alpha_{dist(x_i)}}$$

where $dist(x_i)$ is the distance between the context word x_i and the target word w. The target word being an informative feature present in all samples, we use it in a target word language model P_{Tar}. The surrounding words are summed in the context model P_{Con} as shown in equation (1). As we can see with the presence of α in the equation, the scoring follows the same weighting scheme as we do when accumulating counts, since the samples to classify follow the same distribution as the training ones. Also, when a language model uses automatic Dirichlet smoothing, the impact of the features against the prior is controlled with the manual parameters λ_{Tar} or λ_{Con}. When a manual smoothing parameter is used, it also handles impact control. Our systems use the following weight functions:

Uniform: $\alpha_i = \mathbf{1}_{1 \leq i \leq \delta}$, where δ is a window size and $\mathbf{1}$ the indicator function.

Linear: $\alpha_i = \max\{0, 1 - (i-1)\delta\}$, where δ is the decay rate.

Exponential: $\alpha_i = e^{-(i-1)\delta}$, where δ is the exponential parameter.

Learned: α_i is the weight learned as shown previously.

The parameters for NB systems are identical for all words of a task and were selected by exhaustive search, maximizing leave-one-out accuracy on the training set. For each language model, we tried Laplace, manual Dirichlet and automatic Dirichlet smoothing.

For the sake of comparison, also we provide a *Support Vector Machine* (SVM) classifier, which produces the best results in Semeval 2007. We used libSVM with a linear kernel, and regularization parameters were selected via grid search maximizing leave-one-out accuracy on the training set. We tested the following windows limits: all words in sample, current sentence, and various fixed window sizes. We used the same features as the NB systems, testing Boolean, raw count, log-of-counts and counts from weight functions representations. Although non-Boolean features had good leave-one-out precision on the training data, since SVM does not employ smoothing, only Boolean features kept good results on test data, so our SVM baseline uses Boolean features.

5 WSD experiments on Semeval-2007 English Lexical Sample

The Semeval workshop holds WSD tasks such as the English Lexical Sample (ELS) (Pradhan et al., 2007). The task is to maximize WSD accuracy on a selected set of polysemous words, 65 verbs and 35 nouns, for which passages were taken from the WSJ Tree corpus. Passages contain a couple of sentences around the target word, which is manually annotated with a sense taken from OntoNotes (Hovy et al., 2006). The sense inventory is quite coarse, with an average of 3.6 senses per word. Instances count are listed in Table 1.

	Train	Test	Total
Verb	8988	2292	11280
Noun	13293	2559	15852
Total	22281	4851	

Table 1: Number of instances in the ELS data

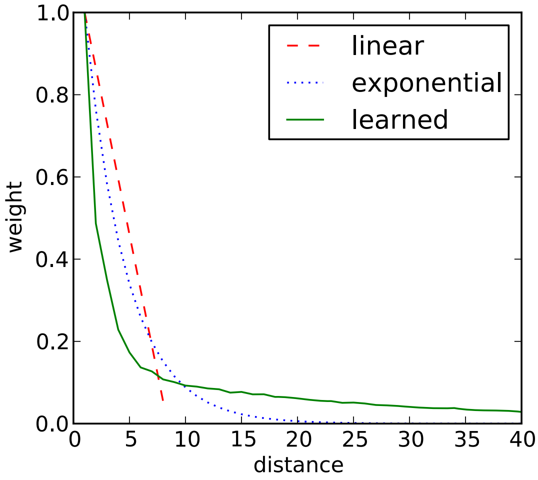


Figure 1: Weight curve for AP88-90

Since there are only 100 target words and instances are limited in the Semeval collection, we do not have sufficient samples to estimate the optimal weights for context words. Therefore, we used the AP88-90 corpus of the TREC collection (CD 1 & 2) in our training process. The AP collection contains 242,918 documents. Since our classifiers use word stems, the collection was also stemmed with the Porter stemmer and sets of windows were built for all word stems. To get near-uniform counts in all distances, only full windows with a size of 100, which was considered big enough without any doubt, were kept. In order to get more samples, windows to the right and to the left were separated. For each target word, we used 1000 windows. A stoplist of the top 10 frequent words was used, but place holders were left in the windows to preserve the distances. Multiple consecutive stop words (ex: "of the") were merged, and the target word stem, being the same for all samples of a set, was ignored in the construction of context models. The AP collection results in 32,650 target words containing 5,870,604 windows. The training process described in Section 3 is used to determine the best weights of context words. Figure 1 shows the first 40 elements of the resulting weighting function curve.

As we can see, the curve is neither exponential, linear, or any of the forms used by Lv and Zhai. Its form is rather similar to $x^{-\delta}$, or rather $\log^{-1}(\delta + x)$ minus some constant. The decrease

System	Cross-Val (%)	Test set (%)
MFS	78.66	77.76
Uniform NB	86.04	84.52
SVM	85.53	85.03
Linear NB	86.89	85.71
Exp. NB	87.80	86.23
Learned NB	88.46	86.70

Table 2: WSD accuracy on Semeval-2007 ELC

rate is initially very high and then reduces as it becomes closer to zero. This long tail is not present in any of the previously suggested functions. The large difference between the above optimal weighting function and the functions used in previous studies would indicate that the latter are suboptimal. Also, as we can see, the relation between context words and the target word is mostly gone after a few words. This would motivate the commonly used very small windows when using a uniform weights, since using a bigger window would further widen the gap between the used weight and the optimal ones.

Now for the system settings, the context words were processed the same way as the external corpus. The target word was used without stemming but had the case stripped. The NB systems used the concatenation of the AP collection and the Semeval data for the collection language model. This is motivated by the fact that the Semeval data is not balanced: it contains only a small number of passages containing the target words. This makes words related to them unusually frequent. The class priors used an absolute discounting of 0.5 on class counts. *Uniform NB* uses a window of size 4, a Laplace smoothing of 0.65 on P_{Tar} and an automatic Dirichlet with $\lambda_{Con} = 0.7$ on P_{Con}. *Linear NB* has $\delta = 0.135$, uses a Laplace smoothing of 0.85 on P_{Tar} and an automatic Dirichlet with $\lambda_{Con} = 0.985$ on P_{Con}. *Exp NB* has $\delta = 0.27$, uses a Laplace smoothing of 2.8 on P_{Tar} and an automatic Dirichlet with $\lambda_{Con} = 1.01$ on P_{Con}. The *SVM* system uses a window of size 3. Our system, *Learned NB* uses a Laplace smoothing of 1.075 on P_{Tar}, and an automatic Dirichlet with $\lambda_{Con} = 1.025$ on P_{Con}. The results on WSD are listed in Table 2. WSD accuracy is measured by the proportion of correctly disambiguated words among all the word samples. The cross-validation is performed on the training data with leave-one-out and is shown as a hint of the capacity of the models. A randomization test comparing *Exponential NB* and *Learned NB* gives a p-value of 0.0508, which is quite good considering the extensive trials used to select the exponential parameter in comparison to a single curve computed from a different corpus. This performance is comparable to the current state of the art. It outperforms most of the systems participating in the task (Pradhan et al., 2007). Out of 14 systems, the best results had accuracies of 89.1*, 89.1*, 88.7, 86.9 and 86.4 (* indicates post-competition submissions). Notice that most previous systems used SVM with additional features such as local collocations, positional word features and POS tags. Our approach only uses bag-of-words in a Naïve Bayes classifier. Therefore, the performance of our method is sub-optimal. With additional features and better classification methods, we can expect that better performance can be obtained. In future work, we will investigate the applications of SVM with our new term weighting scheme, together with additional types of features.

6 WSD experiments on Semeval-2010 Japanese Lexical Sample

The Semeval-2010 Japanese WSD task (Okumura et al., 2010) consists of 50 polysemous words for which examples were taken from the BCCWJ corpus (Maekawa, 2008). It was manually segmented, POS-tagged, and annotated with senses taken from the Iwanami Kokugo dictionary. The selected words have 50 samples for both the training and test set. The task is identical to the ELS of the previous experiment.

Since the data was again insufficient to compute the optimal weighting curve, we used the Mainichi-2005 corpus of NTCIR-8. We tried to reproduce the same kind of segmentation as the training data by using the Chasen parser with UniDic, which nevertheless results in different word segments as the training data. For the corpus and Semeval data, conjugations (setsuzoku-to, jodôshi, etc.), particles (all jo-shi), symbols (blanks, kigô, etc.), and numbers were stripped. When a

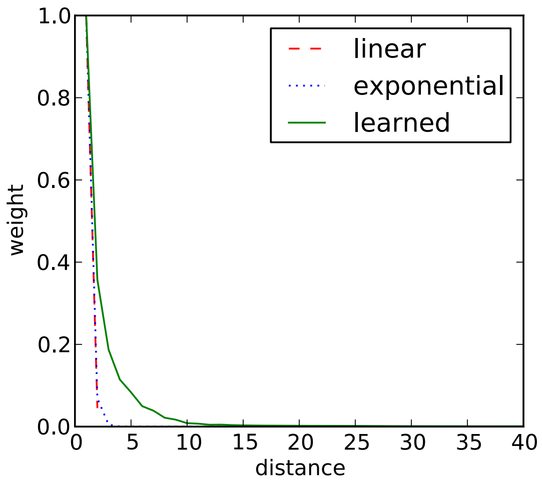


Figure 2: Weight curve for Mainichi 2005

base-form reading was present (for verbs and adjectives), the token was replaced by the Kanjis (Chinese characters) in the word writing concatenated with the base-form reading. This treatment is somewhat equivalent to the stemming+stop list of the ELS tasks. The resulting curve can be seen in Figure 2.

As we can see, the general form of the curve is similar to that of the English collection, but is steeper. This suggests that the meaning of Japanese words can be determined using only the closest context words. Words further than a few positions away have very small impact on the target word. This can be explained by the grammatical structure of the Japanese language. While English can be considered a Subject-Verb-Complement language, Japanese is considered Subject-Complement-Verb. Verbs, mostly found at the end of a sentence, can be far apart from their subject, and vice versa. The window distance is therefore less useful to capture the relatedness in Japanese than in English since Japanese has more non-local dependencies.

The Semeval Japanese test data being part of a balanced corpus, untagged occurrences of the target words are plenty, so we can benefit from using the collection-level counts for smoothing. *Uniform NB* uses a window of size 1, manual Dirichlet smoothing of 4 for P_{Tar} and 90 for the P_{Con}. *Linear NB* has $\delta = 0.955$, uses a manual Dirichlet smoothing of 6.25 on P_{Tar} and manual Dirichlet

System	Cross-Val (%)	Test set (%)
MFS	75.23	68.96
SVM	82.55	74.92
Uniform NB	82.47	76.16
Linear NB	82.63	76.48
Exp. NB	82.68	76.44
Learned NB	82.67	76.52

Table 3: WSD accuracy on Semeval-2010 JWSD

smoothing with $\lambda_{Con} = 65$ on P_{Con}. *Exp NB* has $\delta = 2.675$, uses a manual Dirichlet smoothing of 6.5 on P_{Tar} and a manual Dirichlet of 70 on P_{Con}. The *SVM* system uses a window size of 1 and Boolean features. *Learned NB* used a manual Dirichlet smoothing of 4 for P_{Tar} and automatic Dirichlet smoothing with $\lambda_{Con} = 0.6$ for P_{Con}. We believe this smoothing is beneficial only on this system because it uses more words (the long tail), that makes the estimation of the pseudo-count more accurate. Results on WSD are listed in Table 3. As we can see, the difference between the NB models is less substantial than for English. This may be due to differences in the segmentation parameters of our external corpus: we used the human-checked segmentation found in the Semeval data for classification, but used a parser to segment our external corpus for weight learning. We are positive that the Chasen parser with the UniDic dictionary was used to create the initial segmentation in the Semeval data, but there may be differences in versions and the initial segmentation results were further modified manually.

Another reason for the results could be that the systems use almost the same weights: *Uniform NB* and *SVM* both used windows of size 1, and the Japanese curve is steeper than the English one, making the context model account to almost only immediately adjacent words. So, even if our context model contains more context words at larger distances, their weights are very low. This makes all context model quite similar. Nevertheless, we still observe some gain in WSD accuracy. These results show that the curves work as expected even in different languages. However, the weighting curve is strongly language-dependent. It could also be collection-dependent – we will investigate this aspect in the future, using different collections.

7 Conclusions

The definition of context vector and context model is critical in WSD. In previous studies in IR, decaying weight along with distance within a text window have been proposed. However, the decaying functions are defined manually. Although some of the functions produced better results than the uniform weighting, there is no evidence showing that these functions best capture the impact of the context words on the meaning of the target word. This paper proposed an unsupervised method for finding optimal weights for context words according to their distance to the target word. The general idea was to find the weights that best fit the data, in such a way that the context models for the same target word generated from two random windows samples become similar. It is the first time that this general principle is used for this purpose. Our experiments on WSD in English and Japanese suggest the validity of the principle.

In this paper, we limited context models to bag-of-words features, excluding additional features such as POS-tags. Despite this simple type of feature and the use of a simple Naïve Bayes classifier, the WSD accuracy we obtained can rival the other state-of-the-art systems with more sophisticated features and classification algorithms. This result indicates that a crucial aspect in WSD is the definition of an appropriate context model, and our weighting method can generate more reasonable weights of context words than using a predefined decaying function.

Our experiments also showed that the optimal weighting function is language-dependent. We obtained two different functions for English and Japanese, although their general shapes are similar. In fact, the optimal weighting function reflects the linguistic properties: as dependent words in Japanese can be further away from the target word due to its linguistic structure, the optimal weighting quickly decays, meaning that we can rely less on distant context words. This also shows a limitation of this study: distance is not the sole criterion to determine the impact of a context word. Other factors, such as POS-tag and syntactic dependency, can play an important role in the context model. These additional factors are complementary to the distance criterion and our approach can be extended to include such additional features. This extension is part of our future work.

Another limitation of straight window distance is that all words introduce the same distance, regardless of their nature. In our experiments, to make the distance a more sensible metric, we merged consecutive stop words in one placeholder token. The idea behind this it that some words, such as stop words, should introduce less distance than others. On the opposite, we can easily understand that tokens such as commas, full stops, parentheses and paragraph should introduce a bigger distance than regular words. We could therefore use a *congruence* score for a word, an indicator showing on average how much what comes before is similar to what comes after the word.

Also, we have combined our weighting schema with NB classifier. Other classifiers such as SVM could lead to better results. The utilization of our new weighting schema with SVM is another future work.

Finally, the weights computed with our method has been used in WSD tasks. The weights could be seen as the expected strength of relation between two words in a document according to their distance. The consideration of word relationships in documents and queries is one of the endeavors in current research in IR. The new weighting schema could be easily integrated with a dependency model in IR. We plan to perform such integration in the future.

Acknowledgments

The authors would like to thank Florian Boudin and Satoko Fujisawa for helpful comments on this work. This work is partially supported by Japanese MEXT Grant-in-Aid for Scientific Research on Info-plosion (#21013046) and the Japanese MEXT Research Student Scholarship program.

References

Bai, Jing, Dawei Song, Peter Bruza, Jian-Yun Nie, and Guihong Cao. 2005. Query expansion using term relationships in language models for information retrieval. In *CIKM '05 Proceedings*, pages 688–695, New York, NY, USA. ACM.

Cai, Jun Fu, Wee Sun Lee, and Yee Whye Teh. 2007. Nus-ml: improving word sense disambiguation using topic features. In *SemEval '07 Proceedings*, pages 249–252, Morristown, NJ, USA. Association for Computational Linguistics.

Cheung, Percy and Pascale Fung. 2004. Translation disambiguation in mixed language queries. *Machine Translation*, 18(4):251–273.

Gao, Jianfeng, Ming Zhou, Jian-Yun Nie, Hongzhao He, and Weijun Chen. 2002. Resolving query translation ambiguity using a decaying co-occurrence model and syntactic dependence relations. In *SIGIR '02 Proceedings*, pages 183–190, New York, NY, USA. ACM.

Ide, Nancy and Jean Véronis. 1998. Introduction to the special issue on word sense disambiguation: the state of the art. *Comput. Linguist.*, 24(1):2–40.

Lv, Yuanhua and ChengXiang Zhai. 2009. Positional language models for information retrieval. In *SIGIR '09 Proceedings*, pages 299–306, New York, NY, USA. ACM.

Maekawa, Kikuo. 2008. Compilation of the balanced corpus of contemporary written japanese in the kotonoha initiative (invited paper). In *ISUC '08 Proceedings*, pages 169–172, Washington, DC, USA. IEEE Computer Society.

Navigli, Roberto. 2009. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):1–69.

Okumura, Manabu, Kiyoaki Shirai, Kanako Komiya, and Hikaru Yokono. 2010. Semeval-2010 task: Japanese wsd. In *SemEval '10 Proceedings*. Association for Computational Linguistics.

Pradhan, Sameer S., Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. Semeval-2007 task 17: English lexical sample, srl and all words. In *SemEval '07 Proceedings*, pages 87–92, Morristown, NJ, USA. Association for Computational Linguistics.

Song, D. and P. D. Bruza. 2003. Towards context sensitive information inference. *Journal of the American Society for Information Science and Technology*, 54(4):321–334.

Xu, Jinxi and W. Bruce Croft. 1998. Corpus-based stemming using cooccurrence of word variants. *ACM Trans. Inf. Syst.*, 16(1):61–81.

Zhai, ChengXiang and John Lafferty. 2002. Two-stage language models for information retrieval. In *SIGIR '02 Proceedings*, pages 49–56, New York, NY, USA. ACM.

Measuring the Non-compositionality of Multiword Expressions

Fan Bu and **Xiaoyan Zhu**
State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Sci. and Tech., Tsinghua University
buf08@mails.tsinghua.edu.cn and zxy-dcs@tsinghua.edu.cn

Ming Li
David R. Cheriton School of Computer Science
University of Waterloo
mli@uwaterloo.ca

Abstract

Multiword Expressions (MWEs) appear frequently and ungrammatically in the natural languages. Identifying MWEs in free texts is a very challenging problem.

This paper proposes a knowledge-free, training-free, and language-independent Multiword Expression Distance (MED). The new metric is derived from an accepted physical principle, measures the distance from an n-gram to its semantics, and outperforms other state-of-the-art methods on MWEs in two applications: question answering and named entity extraction.

1 Introduction

A Multiword Expression (MWE) is a sequence of neighboring words "whose exact and unambiguous meaning or connotation cannot be derived from the meaning or connotation of its components" (Choueka, 1988). In the paper, MWEs refer to non-compositional lexical units including idioms, terminologies and name entities. As Jackendoff (1997) notes, the magnitude of MWEs is far greater than what has traditionally been realized within linguistics. He estimates that the number of MWEs in a speaker's lexicon is of the same order of magnitude as the number of single words. In WordNet 1.7 (Fellbaum, 1998), 41 percent of the entries are multi-words. Some specialized domain vocabulary, such as terminology, overwhelmingly consists of MWEs. Automatic extraction of MWEs is indispensable to many tasks such as machine translation, name entity extraction, information retrieval and question answering.

Due to their non-compositionality, many MWEs cannot be directly identified using grammatical rules, which poses a major challenge to automatic analysis. Moreover, existing resources like dictionaries can never have adequate and timely coverage. Therefore people turn to statistical method to characterize MWEs.

Since Church and Hanks (1990) proposed Pointwise Mutual Information (PMI), a variety of measures, such as Log-likelihood, Symmetrical Conditional Probability (SCP) and Mutual Expectation (Dias et al., 2000), have been introduced to measure word association. Their basic ideas are very similar: the whole n-gram is separated into two parts and the association is determined by the joint probability and the probability of each part. Pecina (2006) compared 84 bi-gram association measures and found PMI has the best performance in Czech data. When applying these measures to the n-grams for $n > 2$, it is not clear how can the association between the deliberately separated two parts represent the non-compositionality of the whole n-gram. Different policies have been studied to extend these measures into arbitrary n-grams (Silva and Lopes, 1999; Schone and Jurafsky, 2001; Dias et al., 2000). Is there a fundamental, less arbitrary, and general approach to this problem? That is,

- Can we actually derive a MWE metric for n-grams from the first principles, instead of making a seemingly sensible, but really arbitrary, proposal?
- Will such a theoretically justified new metric actually works better than other heuristic

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 116–124,
Beijing, August 2010

measures for general MWEs?

This paper will answer above questions positively. We derive an optimal distance metric Multiword Expression Distance (MED). MED defines the semantic function for n-grams and the information distance (Bennett et al., 1998) from the n-grams to their semantics. Unlike previous methods it ensures the cohesion of the n-gram directly hence applicable to MWEs of any length.

The MED is naturally generalized to its conditional version. The extension is based on the observation that many MWEs are domain dependent. It is true that some MWEs are only used in certain domains, but they are domain free. For example, we know that "polymerase chain reaction" is some sort of terminology even if many of us do not know what it is exactly. However that is not always the case. For those who do not watch movies, the sentence "catch me if you can" will probably be taken as a non-MWE, instead of a movie name. The non-compositionality of this sentence appears only in the movies domain. The experimental results show that given appropriate phrases as conditions, the conditional MED performs better than MED.

We also investigate the efficacy of MED on post-processing of Question Answering (QA) and complex named entity extraction. The experimental results show that our method outperforms state of art methods (Zhang et al., 2009; Downey et al., 2007) in these two applications. Moreover, MED is a pure statistical metric which can be easily combined with other methods.

The remainder of this paper is organized as follows: In the next section we review the related work on Multiword Expression and information distance. Section 3 gives a preliminary introduction to Kolomogorov complexity and information distance. Section 4 proposes the formal definition of MED. In Section 5 we discuss the difference between MED and Pointwise Mutual Information. We apply MED to QA post-processing and complex named entity extraction in Section 6 and evaluate their performance in Section 7. In the last section we conclude this work.

2 Related Work

Researchers have explored various techniques for identifying MWEs. These approaches could be broadly classified into three types: linguistic methods, sequential tagging based methods and statistical methods.

The mostly used linguistic information for MWE extraction is words' Part-Of-Speech tags. Justeson and Katz (1995) extracted technical terminologies from documents using a regular expression on POS-tags of a word sequence, together with some frequency constraints. Argamon et al. (1998) separated the POS sequence of a multi-word into small POS tiles, counted tile frequency in the MWE and non-MWE training sets and identify new MWEs by these counts. Although linguistic methods perform well in term extraction on specific domains, it cannot be generalized to identify arbitrary MWEs.

Several supervised learning methods have been used previously for extracting Name Entities including Hidden Markov Models, Maximum Entropy Markov Models and Conditional Random Field (CRF) models (McCallum and Li, 2003). In order to allow tractable computation, these models can only use local features in a small window. Although the approximate inference methods have been incorporated into sequential tagging model to capture non-local information (Finkel et al., 2005), these models are not capable of recognizing complex named entities, especially those containing conjunctions and prepositions. Experimental results in (Downey et al., 2007) show that statistical methods substantially outperform sequential tagging based methods on identifying complex named entities.

In statistical methods for MWE extraction, Church and Hanks (1990) first presented Pointwise Mutual Information (PMI) as an objective measure for estimating word association. Since then, many methods has been proposed to measure bi-gram association, such as Log-likelihood (Dunning, 1993) and Symmetrical Conditional Probability (Silva and Lopes, 1999). Pecina (2006) compared 84 bi-gram association measures and concluded that PMI had the best performance in Czech data. When it comes to measure the non-compositionality for arbitrary n-grams, policies were taken to separate n-gram into two parts X and Y so that it can be measured by existing bi-gram methods (such as PMI). Silva and Lopes (1999) and Dias et al. (2000) calculated the arithmetic average of every possible separation. Schone and Jurafsky (2001) define X and Y to be the word sequences $w_1 w_1 ... w_i$ and $w_{i+1} w_{i+2} ... w_n$, where i is chosen to maximize $P_x P_y$. Recently Zhang et al. (2009) proposed Enhanced Mutual Information (EMI) which measured the cohesion of n-gram by the frequency of itself and the frequency of each word.

The information distance is a universal distance measure between two information carrying entities (Bennett et al., 1998; Li et al., 2001; Li et al., 2004). The applications of information distance using compression were first introduced in (Li et al., 2001) and then in (Bennett et al., 2003; Chen et al., 2004). The experimental results in (Keogh et al., 2004) showed that information distance/compression based method was superior to 51 parameter-laden methods from seven major data mining conferences on their benchmark data. The web-based approximation of information distance was introduced by Cilibrasi and Vitányi (2007) to measure the semantic similarity of two words or concepts.

3 Preliminaries

3.1 Kolmogorov Complexity

Kolmogorov complexity defines randomness of an individual string. Fix a universal Turing machine U, the *Kolmogorov complexity* of a binary string x condition to another binary string y $K_U(x|y)$ is defined as the length of the shortest (prefix-free) program for U that outputs x with input y. It can be shown that for a different universal Turing machine U', for all x, y

$$K_U(x|y) = K_{U'}(x|y) + C, \tag{1}$$

where the constant C depends only on U'. Thus, we can simply write $K_U(x|y)$ as $K(x|y)$ and $K(x|\epsilon)$ as $K(x)$, where ϵ is the empty string.

3.2 Information Distance

Between any two information carrying entities, is there an objective distance that is application-independent and unique, similar to the concept of distance in the physical world? From a commonly accepted physical principle of von Neumann and Landauer that irreversibly processing one bit of information costs 1KT of energy, Bennett et al. (1998) derived exactly such a distance: the Information Distance. Information Distance $E(x, y)$ between two objects x and y is the energy to convert between x and y. Bennett et al. (1998) proved:

Theorem 1 *Up to an additive logarithmic term,* $E(x, y) = \max\{K(x|y), K(y|x)\}$.

Thus, the max distance was defined below (Bennett et al., 1998):

$$D_{max}(x, y) = \max\{K(x|y), K(y|x)\}.$$

D_{max} was shown to satisfy distance requirements such as positivity, symmetricity and triangle inequality (Bennett et al., 1998). It was further shown that D_{max} is optimal in the sense that it is universal. That is, it minorizes (up to constant factors) all other nontrivial and computable distances. More precisely, a distance D is admissible if

$$\sum_y 2^{-D(x,y)} \leq 1. \tag{2}$$

Thus, we exclude trivial distances such as $d(x, y) = 0$ for all x, y. It was proved in (Bennett et al., 1998) that for any admissible computable distance D, there is a constant c, for all x,y,

$$D_{max}(x, y) \leq D(x, y) + c.$$

In other words, if any such distance D discovers some similarity between x and y, so will D_{max}.

In order to deal with the information carrying objects of different sizes, the normalized information distance was proposed in (Li et al., 2001). In (Li et al., 2004), the normalized max distance was defined as:

$$d_{max}(x, y) = \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}}$$

d_{max} satisfies positivity, symmetricity, triangle inequality and some weak form of universality (Li et al., 2004).

4 A New Metric for MWE

4.1 The Semantics

When applying the Information Distance to identifying MWEs, how to encode n-grams and their semantics is the first thing to be considered. It is inappropriate to encode MWEs literally. For example, when referring to "kick the bucket", the three words "kick", "the" and "bucket" cannot represent all the semantics about this expression.

Inspired by Cilibrasi and Vitányi (2007), we define *context* of an n-gram as the set of all the web pages containing it. Also, *semantic* of an n-gram is defined as the set of all the web pages containing all the words appeared in that n-gram. For example, the semantic of "U.S. president" including not only the pages containing itself but also those containing "the president of U.S." or "president Obama says that ... U.S. government...".

4.2 Multiword Expression Distance

Let us denote the vocabulary set by S and the set of web pages by Ω. The cardinality of Ω is denoted by M=$|\Omega|$. Define $G \equiv S^+$ as the set of n-grams. A search term t is defined as an n-gram or the conjunction of search terms. Denote T as the set of search terms and we have $G \subset T$. Let $\phi : T \to 2^\Omega$ be the *context* function mapping each search term t to the web set which includes (and only includes) all the web pages containing all the n-grams in t. Let $\theta : G \to T$ be the function mapping each n-gram $g = w_1w_2...w_n$ to $\bigwedge_i w_i$, the conjunction of the words in it. Finally we define the *semantic* function $\mu : G \to 2^\Omega$ as the composite function $\phi \circ \theta$. It is obvious that for any n-gram g, we have $\phi(g) \subseteq \mu(g)$. Given an n-gram g, we will encode $\phi(g)$ and $\mu(g)$ and calculate the distance between them.

While $K(x)$ is not computable, a simple heuristic, noticed by Cilibrasi and Vitányi (2007), is to use Shannon-Fano code to encode the probability (approximated by its internet frequency) of x. Assume that all web pages are equiprobable, with the probability of being returned by search engine being $\frac{1}{M}$. Let $p : \phi(T) \to [0, 1]$ be the *context* probability function in which $\phi(T) \equiv \{x|\exists y \in T, x = \phi(y)\}$. Since each context is a set of webpages, the probability of context c is defined as $p(c) = \frac{|c|}{N}$ where $N = \sum_{c\in\phi(T)} |c|$ ensures p is a valid probability function. The Shannon-Fano code (Li and Vitányi, 2008) length associated with p can then be regarded as an approximation of K,

$$K(x) \approx -\log p(x) \quad (3)$$

$$K(x, y) \approx -\log p(x, y) \quad (4)$$

According to (3),(4) and Theorem 1, D_{max} can be approximated as follows:

$$\begin{aligned} & D_{max}(x, y) \\ = \; & \max\{K(x|y), K(y|x)\} \\ = \; & K(x, y) - \min\{K(y), K(x)\} \\ \approx \; & \max\{\log|x|, \log|y|\} - \log|x \cap y| \end{aligned}$$

Similarly, we have

$$\begin{aligned} & D_{max}(x, y|c) \\ \approx \; & \max\{\log|x \bigcap c|, \log|y \bigcap c|\} - \log|x \bigcap y \bigcap c| \end{aligned}$$

Since $\phi(g) \subseteq \mu(g)$, the Multiword Expression Distance of an n-gram g can be defined as follows:

$$\begin{aligned} & \mathrm{MED}(g) \\ \equiv \; & D_{max}(\phi(g), \mu(g)) \\ \approx \; & \max\{\log \tfrac{|\phi(g)|}{|\phi(g) \bigcap \mu(g)|}, \log \tfrac{|\mu(g)|}{|\phi(g) \bigcap \mu(g)|}\} \\ = \; & \log|\mu(g)| - \log|\phi(g)| \end{aligned}$$

Given a search term c as condition, the Conditional Multiword Expression Distance of an n-gram g is defined as follows:

$$\begin{aligned} \mathrm{MED}(g|c) \; \equiv \; & D_{max}(\phi(g), \mu(g)|\phi(c)) \\ \approx \; & \log|\mu(g) \bigcap \phi(c)| - \log|\phi(g) \bigcap \phi(c)| \end{aligned}$$

Based normalized information distance, NMED and its conditional version can be derived as follows:

$$\mathrm{NMED}(g) \approx \frac{\log|\mu(g)| - \log|\phi(g)|}{\log N - \log|\phi(g)|}$$

$$\mathrm{NMED}(g|c) \approx \frac{\log|\mu(g) \bigcap \phi(c)| - \log|\phi(g) \bigcap \phi(c)|}{\log|\phi(c)| - \log|\phi(g) \bigcap \phi(c)|}$$

Where N can be estimated from the size of internet by some combinatorial methods.

To implement MED by a general search engine, we assume Ω to be the set of indexed webpages. Thus, $|\phi(g)|$ and $|\mu(g)|$ can be approximated by the hit numbers given g and the "logic and" of each word in g as queries. Yahoo Search is used in our experiments.

5 Relation with Pointwise Mutual Information

When $n = 2$, we denote $\mathrm{P}(w_1 w_2)$ the probability of a web page containing bi-gram $g = w_1 w_2$ and $\mathrm{P}(w_1 \bigwedge w_2)$ the probability of a web page containing w_1 and w_2. Assuming the occurrence of w_1 and w_2 are independent, we have

$$\begin{aligned} \mathrm{MED}_2(g) &= \log \frac{|\phi(w_1 \bigwedge w_2)|}{|\phi(w_1 w_2)|} \\ &= \log \frac{\mathrm{P}(w_1 \bigwedge w_2)}{\mathrm{P}(w_1 w_2)} \\ &\approx \log \frac{\mathrm{P}(w_1) P(w_2)}{\mathrm{P}(w_1 w_2)} \\ &\propto -\mathrm{PMI}(g) \end{aligned}$$

Thus, PMI is inversely proportional to MED under the independence assumption. This assumption is unadvisable for obvious reasons. PMI compares the probability of observing x and y within a given window w (w=2 when measuring collocation) with the probabilities of observing x and y independently. However, most of the word sequences in practice (both MWEs and non-MWEs) are far from being independent. Therefore the assumption potentially creates additional noises to MED, especially when $n > 2$. The internet contains billions of pages and thus we can count the pages containing specified words directly without making independent assumption to overcome data sparseness.

6 Applications

6.1 MWE for QA Systems

Some types of questions require a QA system to return phrases as the answers instead of sentences, such as Factoid and List. Given a question, we need to generate queries, obtain relevant pages from the internet, extract the candidate n-grams from relevant pages and finally rank all the candidates by their likelihood of being an answer.

Some previous work exploited web redundancy to estimate answer validity(Magnini et al., 2002; Zhang et al., 2008). No research, to our knowledge, has focused on checking the completeness of candidates. Most of texts on the internet are informal (e.g. they contain uncapitalized proper nouns and incomplete sentence structures). Parser and named entity recognizers trained on formal corpus are unpractical on recognize NP chunks or name entities on the web.

Observing that each candidate is n-gram and checking the completeness of a candidate is to measure its non-compositionality, we introduce a simple MWEs-based method to rank all candidates by their completeness and merge similar answers.

Given a question and a list of candidate answers:

1. Extract proper nouns from the question as conditions.
2. Calculate the conditional MED (or MED if no proper noun is found in question) for each candidate. Then for each pair of literally similar candidates, the one with larger MED distance is removed.
3. Rank the rest candidates by conditional MED.

This method is case insensitive and do not rely on context information. All of the statistics are performed on the internet thus no local corpus is needed.

6.2 Complex Named Entity Extraction

In many previous work (McCallum and Li, 2003; Finkel et al., 2005), named entity extraction is combined with classification, which is known as Name Entity Recognition (NER). Most of these NER technique are based on sequential tagging models and unsuitable to the task of locating complex named entities in Web text. In (Downey et al., 2007), the author treated named entity as a type of MWE and proposed the algorithm LEX++ to locate complex named entities.

Inspired by Downey's work, we propose a conditional MED based algorithm MWE++ to extract named entities. Given a sentence $S = \{S_1, S_2 ... S_n\}$ and parameters τ_1,τ_2 and δ, MWE++ proceeds as follows:

1. Initialize a sequence of names $N = (n_1, n_2, ..., n_M)$ equal to the maximal contiguous substrings of S that consist entirely of capitalized words. If the first word of S appears capitalized in the local corpus and it is at the beginning of a sentence more than δ of the times, it is omitted from N.

2. Until N does not change during last iteration:

   (a) Choose the *mergeable* pair of names (n_i, n_{i+1}) with minimum conditional MED.

   (b) Replace n_{min_i} and n_{min_i+1} with the single name $n_{min_i} w_{min_i} n_{min_i+1}$ where w_i is the uncapitalized words between n_i and n_{i+1}.

3. For every names n_i in N

   (a) Check common prefix and punctuation at boundary of n_i via local corpus.

   (b) Check number at boundary of n_i via internet.

In MWE++, We define two thresholds τ_1 and τ_2 to estimate the name entity confidence of a given n-gram. If $\mathrm{MED}(g|.)$ is lower than τ_1, between τ_1 and τ_2 or higher than τ_2, $\mathrm{conf}(g)$ will be 2 (Definitely), 1 (Probably) or 0 (Impossible). The confidence of all initialized capitalized words will be set to 1. If an n-gram contain unmatched brackets or quotation marks, its confidence will be set to 0. Also, The confidence of n-gram containing comma will be reduced by 1. We say a pair of names (n_i, n_{i+1}) is mergeable if and only if $\mathrm{conf}(n_i w_i n_{i+1}) \geq \max(\mathrm{conf}(n_i), \mathrm{conf}(n_{i+1}))$.

After iteration, we will check common prefixes, punctuations and numbers at boundary of each names. If a name n_i is immediately preceded by a single number t and $\mathrm{conf}(tn_i) \geq 1$, we replace n_i by tn_i. Similarly, a number t immediately following n_i is appended to n_i when $\mathrm{conf}(n_i t) \geq 1$. Due to the limitation of search engine, punctuation check and common prefix check modules are performed on local corpus just the same as LEX++.

7 Experiments and Analysis

7.1 Compositionality Measure

In this section, we evaluate how well can MED separate non-compositional phrases (idioms) from compositional ones. First we evaluate MED and other four metrics on English_VPC data published on the MWE 2008 shared task. The data set contains 3078 verb-noun bi-grams and 14 percent of them are annotated as idiomatic. The average precision of MED, PMI, SCP, t-score and EMI (Zhang et al., 2009) are 0.234, 0.233, 0.285, 0.274 and 0.205. The result shows that MED is not distinguished on bi-grams test. It is partly because most idiomatic verb-noun collocations are often used non-idiomatically. Their compositionality are not necessarily lower than non-idiomatic ones.

We also evaluate different metrics on n-grams of varied lengths. Since all published MWE data sets we find only contain bi-grams, we construct our test set as follows. We first collected common idioms from the lists of english idioms on Wikipedia. To get enough common but not idiomatic phrases, we collect common compositional phrases from UsingEnglish.com, englishspeak.com, Wikipedia and China Daily BBS. Since it is difficult for non-native speakers to pick up idioms from non-idiomatic ones, we do not manually check all compositional phrases. The test set contains 1529 idioms and 1798 compositional phrases. The n-gram frequencies are not significantly different between idioms and compositional phrases. The mean and standard deviation are 2.1×10^5 and 7.8×10^5 on idioms and 7.4×10^5 and 4.8×10^6 on compositional phrases. We employ different measures to rank all the phrases. Non-conditional MED and NMED are compared with AVG_SCP (Silva and Lopes, 1999), MAX_PMI (Schone and Jurafsky, 2001), EMI (Zhang et al., 2009) and the baseline n-gram frequency. T-score is not under evaluation because we do not find sound n-gram extension for it. The precision-recall curve is shown in Fig. 1. Since the performance of MED and NMED are very

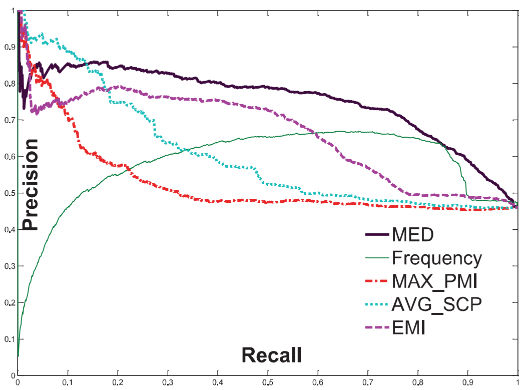


Figure 1: Precision-recall curves of five measures

	freq	MAX_PMI	AVG_SCP	EMI	NMED	MED	MED(.|.)
fairy tale	0.493	0.484	0.570	0.515	0.615	0.617	**0.657**
science fiction	0.500	0.470	0.558	0.525	0.596	0.599	**0.633**
action movie	0.695	0.523	0.723	0.703	0.763	0.768	**0.823**
animation	0.561	0.642	**0.693**	0.489	0.671	0.673	0.689
horror movie	0.595	0.528	0.647	0.633	0.667	0.670	**0.692**
documentary	0.525	0.549	0.626	0.512	0.596	0.598	**0.654**
hip hop	0.598	0.627	0.645	0.635	0.652	0.651	**0.712**
jazz	0.549	0.501	0.543	0.539	0.627	0.625	**0.716**
rock&roll	0.742	0.567	0.730	0.741	0.708	0.717	**0.836**
company	0.614	0.584	0.689	0.663	0.754	**0.756**	0.735
soccer player	0.945	0.648	0.904	**0.973**	0.911	0.918	0.941
novelists	0.772	0.701	**0.870**	0.866	0.821	0.828	0.864
PS3 game	0.603	0.675	0.740	0.535	0.742	**0.744**	0.727
overall	0.612	0.577	0.688	0.629	0.696	0.700	**0.726**

Table 1: Performance of different measures in each list

close, NMED is not displayed for clarity. From the result we can see that MED performs substantially better than all the other measures. Average precision(avp) of the top 3 measures MED, EMI and AVG_SCP are 0.75, 0.71 and 0.66.

7.2 QA Post-processing

It is difficult to evaluate the method introduced in Section 6.1 directly since QA benchmarks mainly focus on accuracy of the top one answer instead of the completeness of top-n candidates. Therefore, the experiment is designed as follows. We extract name lists on different domains from Wikipedia. For each name in each list, we put it into a search engine and get the context from a random selected snippet. For each name, We created two incomplete names by randomly adding (or removing) one or two words according to its context. It is guaranteed that the original name and its counterpart with noise must have at least two words in common. We tag the original names and the noise added ones in each list as positive and negative samples. A list can be regarded as the candidates and the list name (or its synonym) can be seen as the key phrase extracted from question.

The test set can be divided into six common categories: movie, book, music, person, organization and video game. Each category contains one to four lists. The test set contains 11080 samples in total. Still, we employ the measures in previous experiments to rank all the candidates to see if the complete names can be separated from the incomplete names. The results are listed in Table 1. The overall avp is the average of the avp of each lists weighted by their size.

It is shown that the performance of conditional MED is the best over all metrics, followed by MED. The reason why EMI and AVG_SCP get best results on soccer player and novelists is that they take more advantage of frequency. Since the length of people's name are short (2 to 3 words), most of negative samples are created by adding words, which makes frequency important.

7.3 Complex Named Entity Extraction

In this section we evaluate the named entity extraction performance of Algorithm MWE++. The experiment is done on the corpus, the training set and the test set provided by Downey et al. (2007). Four classes of entities (Actor, Book, Company and Film) were manually annotated on both training and test set. All sentences in the corpus contain named entities from the above four classes (but not annotated). The corpus consists of 183,726 sentences while the training and the test set contain 200 and 629 sentences, respectively. Furthermore, test sentences are separated into 100 difficult cases and 529 easy cases. All difficult cases contain complex name entities (entities containing uncapitalized words), such as "Procter and

Gamble" and "Gone with the Wind".

The conditional MED metric in this experiment is redefined as follows:

$$\mathrm{MED}(g|C) = \min_{c \in C}\{\mathrm{MED}(g|c)\},$$

where C={"IMDB","Amazon","corporation"}. "IMDB" is used as the condition of Actor and Film while "Amazon" and "corporation" are chosen to be the condition of Book and Company. We compute the conditional MED for all entities on training set. τ_1 is set to the median and τ_2 is set to the value larger than 90% entities on training set. δ is set to 0.5. MWE++ is performed on the 100 difficult cases. The results shown in Table 2 convincingly show that MWE++ significantly outperforms LEX++, supervised models (SVMCMM, CRF) and rule-based model (MAN) on identifying complex named entities. Compared to LEX++, MWE++ is not only more accurate but also more flexible. LEX++ relies on local corpus while MWE++ does not. When recognizing new entities, we just need to find appropriate condition words instead of preparing new corpus. For the sake of completeness, the F-score of MWE++ on easy cases is 91, which is lower than all the other methods. However this is irrelevant since this part can be made quite accurate by specialized databases and training by any known methods.

All test data in this paper can be downloaded from http://60.195.250.61:8080/download/.

8 Conclusion

We have derived an MWE metric MED from the first principles via Information Distance. The new metric measures the distance from an n-gram to its semantics. It is provably optimal (universal), overcomes several deficiencies of previous approaches, and convincingly outperforms the other methods.

	F_1	Recall	Precision
MAN	0.18	0.22	0.16
CRF	0.35	0.42	0.31
SVMCMM	0.42	0.48	0.37
LEX++	0.74	0.76	0.72
MWE++	**0.83**	**0.86**	**0.80**

Table 2: Named entity extraction on difficult cases

Also, we have taken advantage of the fact that some MWEs are domain dependent. This feature is important when recognizing named entities and terminologies. The conditional MED is better than MED when we know what we are looking for. Since MED is quite different from previous measures, it can be combined with others by machine learning approaches and enhance the overall performance. Further experiments are needed.

Acknowledgment

This work was supported mainly by Canada's IDRC Research Chair in Information Technology program, Project Number: 104519-006. It is also supported by the Chinese Natural Science Foundation grant No. 60973104, NSERC Grant OGP0046506, 863 Grant 2008AA02Z313 from China's Ministry of Science and Technology, Canada Research Chair program, MITACS, an NSERC Collaborative Grant, and Ontario's Premier's Discovery Award.

References

Shlomo Argamon, Ido Dagan and Yuval Krymolowski 1998. A memory-based approach to learning shallow natural language patterns. In *Proc. of COLING*, 1998, pp. 67-73.

Charles H. Bennett, Peter Gács, Ming Li, Paul M.B. Vitányi, and Wojciech H. Zurek 1998. Information distance. *IEEE Trans-IT* 44:4, 1998, pp. 1407-1423.

Charles H. Bennett, Ming Li and Bin Ma 2003. Chain letters and evolutionary histories. *Scientific American*, 288:6, (feature article), 76-81.

Xin Chen, Brent Francia, Ming Li, Brian Mckinnon, Amit Seker 2004. Shared information and program plagiarism detection. *IEEE Trans. Information Theory*, 50:7, 1545-1550.

Yaacov Choueka 1988. Looking for needles in a haystack or locating interesting collocation expressions in large textual databases In *Proc. of the RIAO*, 1988, pp. 38-43.

Kenneth W. Church and Patrick Hanks 1990. Word association norms, mutual information and lexicography. *Computational Linguistics*, Vol. 16, No. 1, pp. 22-29.

Rudi L. Cilibrasi and Paul M.B. Vitányi 2007. The Google similarity distance. *IEEE Trans-Knowledge and Data Engineering* 19:3, 2007, pp. 370-383.

Joaquim F. da Silva and Gabriel P. Lopes 1999. A local maxima method and a fair dispersion normalization for extracting multi-word units from corpora. In *Proc. of Sixth Meeting on Mathematics of Language*, pp. 369-381.

Gaël Dias, Sylvie Guilloré and José G.P. Lopes 2000. Mining textual associations in text corpora. In *Sixth ACM SIGKDD, Workshop on Text Mining*, pp. 92-95.

Ted Dunning 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*. Vol. 19, No. 1, 61-74.

Doug Downey, Matthew Broadhead and Oren Etzioni 2007. Locating complex named entities in web text. In *Proc. of IJCAI*, 2007, pp. 2733-2739.

Christine Fellbaum 1998. WordNet: an electronic lexical database. MIT Press, Cambrige, MA.

Jenny R. Finkel, Trond Grenager and Christopher Manning 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proc. of ACL*, 2005, pp. 363-370.

Ray Jackendoff 1997. The architecture of the language faculty. MIT Press, Cambrige, MA.

John S. Justeson and Slava M. Katz 1995. Technical terminology: Some linguistic properties and an algorithm for indentification in text. *Natural Language Engineering* 1:1, 1995, 9-27.

Eamonn J. Keogh, Stefano Lonardi and Chotirat A. Ratanamahatana 2004. Towards parameter-free data mining. In *Proc. of ACM SIGKDD*,2004, pp. 206-215.

Ming Li, Jonathan H. Badger, Xin Chen, Sam Kwong, Paul Kearney, and Haoyong Zhang 2001. An information-based sequence distance and its application to whole mitochondrial genome phylogeny. *Bioinformatics*, 17:2(2001), 149-154.

Ming Li, Xin Chen, Xin Li, Bin Ma and Paul M.B. Vitányi 2004. The similarity metric. *IEEE Trans-IT* 50:12, 2004, 3250-3264.

Ming Li and Paul M.B. Vitányi 2008. An introduction to kolmogorov complexity and its applications. Springer-Verlag, New York, 2008. Third edition.

Andrew McCallum and Wei Li 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proc. of the 7th conference on Natural language learning at HLT-NAACL*,2003 ,pp. 188-191.

Bernardo Magnini, Matteo Negri and Hristo Tanev 2002. Is it the right answer? Exploiting web redundancy for answer validation. In *Proc. of ACL*,2002, pp. 425-432.

Pavel Pecina 2006. An extensive empirical study of collocation extraction methods. In *Proc. of COLING-ACL*, 2006, pp. 953-960.

Patrick Schone and Daniel Jurafsky 2001. Is knowledge-free induction of multiword unit dictionary headwords a solved problem? In *Proc. of EMNLP, 2001*.

Wen Zhang, Taketoshi Yoshida, Xijin Tang and Tu-Bao Ho 2009. Improving effectiveness of mutual information for substantival multiword expression extraction. *Expert Systems with Applications*, Volume 36, 10919-10930, Elsevier.

Xian Zhang, Yu Hao, Xiaoyan Zhu and Ming Li 2007. New information measure and its application in question answering system. *J. Comput. Sci. Tech.*, 23:4(2008), pp. 557-572. (Preliminary version appeared in SIGKDD 2007.)

A Utility-Driven Approach to Question Ranking in Social QA

Razvan Bunescu
School of EECS
Ohio University
bunescu@ohio.edu

Yunfeng Huang
School of EECS
Ohio University
yh324906@ohio.edu

Abstract

We generalize the task of finding question paraphrases in a question repository to a novel formulation in which known questions are ranked based on their utility to a new, reference question. We manually annotate a dataset of 60 groups of questions with a partial order relation reflecting the relative utility of questions inside each group, and use it to evaluate meaning and structure aware utility functions. Experimental evaluation demonstrates the importance of using structural information in estimating the relative usefulness of questions, holding the promise of increased usability for social QA sites.

1 Introduction

Open domain Question Answering (QA) is one of the most complex and challenging tasks in natural language processing. While building on ideas from Information Retrieval (IR), question answering is generally seen as a more difficult task due to constraints on both the input representation (natural language questions vs. keyword-based queries) and the form of the output (focused answers vs. entire documents). Recently, community-driven QA sites such as Yahoo! Answers and WikiAnswers have established a new approach to question answering in which the burden of dealing with the inherent complexity of open domain QA is shifted from the computer system to volunteer contributors. The computer is no longer required to perform a deep linguistic analysis of questions and generate corresponding answers, and instead acts as a mediator between users submitting questions and volunteers providing the answers. In most implementations of community-driven QA, the mediator system has a well defined strategy for enticing volunteers to post high quality answers on the website. In general, the overall objective is to minimize the response time and maximize the accuracy of the answers, measures that are highly correlated with user satisfaction. For any submitted question, one useful strategy is to search the QA repository for similar questions that have already been answered, and provide the corresponding ranked list of answers, if such a question is found. The success of this approach depends on the definition and implementation of the question-to-question similarity function. In the simplest solution, the system searches for previously answered questions based on exact string matching with the reference question. Alternatively, sites such as WikiAnswers allow the users to mark questions they think are rephrasings (“alternate wordings”, or paraphrases) of existing questions. These question clusters are then taken into account when performing exact string matching, therefore increasing the likelihood of finding previously answered questions that are semantically equivalent to the reference question. Like the original question answering task, the solution to question rephrasing is also based on volunteer contributions. In order to lessen the amount of work required from the contributors, an alternative solution is to build a system that automatically finds rephrasings of questions, especially since question rephrasing seems to be computationally less demanding than question answering. The question rephrasing subtask has spawned a diverse set of approaches. (Herm-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 125–133,
Beijing, August 2010

jakob et al., 2002) derive a set of phrasal patterns for question reformulation by generalizing surface patterns acquired automatically from a large corpus of web documents. The focus of the work in (Tomuro, 2003) is on deriving reformulation patterns for the interrogative part of a question. In (Jeon et al., 2005), word translation probabilities are trained on pairs of semantically similar questions that are automatically extracted from an FAQ archive, and then used in a language model that retrieves question reformulations. (Jijkoun and de Rijke, 2005) describe an FAQ question retrieval system in which weighted combinations of similarity functions corresponding to questions, existing answers, FAQ titles and pages are computed using a vector space model. (Zhao et al., 2007) exploit the Encarta logs to automatically extract clusters containing question paraphrases and further train a perceptron to recognize question paraphrases inside each cluster based on a combination of lexical, syntactic and semantic similarity features. More recently, (Bernhard and Gurevych, 2008) evaluated various string similarity measures and vector space based similarity measures on the task of retrieving question paraphrases from the WikiAnswers repository.

According to previous work in this domain, a question is considered a rephrasing of a reference question Q_0 if it uses an alternate wording to express an identical information need. For example, Q_0 and Q_1 below may be considered rephrasings of each other, and consequently they are expected to have the same answer.

Q_0 What should I feed my turtle?

Q_1 What do I feed my pet turtle?

Community-driven QA sites are bound to face situations in which paraphrasings of a new question cannot be found in the QA repository. We believe that computing a ranked list of existing questions that partially address the original information need could be useful to the user, at least until other users volunteer to give an exact answer to the original, unanswered reference question. For example, in the absence of any additional information about the reference question Q_0, the expected answers to questions Q_2 and Q_3 above may be seen as partially overlapping in information content with the expected answer for the reference question. An answer to question Q_4, on the other hand, is less likely to benefit the user, even though it has a significant lexical overlap with the reference question.

Q_2 What kind of fish should I feed my turtle?

Q_3 What do you feed a turtle that is the size of a quarter?

Q_4 What kind of food should I feed a turtle dove?

In this paper, we propose a generalization of the question paraphrasing problem to a question ranking problem, in which questions are ranked in a partial order based on the relative information overlap between their expected answers and the expected answer of the reference question. The expectation in this approach is that the user who submits a reference question will find the answers of the highly ranked question to be more useful than the answers associated with the lower ranked questions. For the reference question Q_0 above, the system is expected to produce a partial order in which Q_1 is ranked higher than Q_2, Q_3 and Q_4, whereas Q_2 and Q_3 are ranked higher than Q_4. In Section 2 we give further details on the question ranking task and describe a dataset of questions that have been manually annotated with partial order information. Section 3 presents a set of initial approaches to question ranking, followed by their experimental evaluation in Section 4. The paper ends with a discussion of future work, and conclusion.

2 A Partially Ordered Dataset for Question Ranking

In order to enable the evaluation of question ranking approaches, we created a dataset of 60 groups of questions. Each group consists of a reference question (e.g. Q_0 above) that is associated with a partially ordered set of questions (e.g. Q_1 to Q_4 above). The 60 reference questions have been selected to represent a diverse set of question categories from Yahoo! Answers. For each reference questions, its corresponding partially ordered set is created from questions in Yahoo! Answers

REFERENCE QUESTION (Q_r)
Q_5 What's a good summer camp to go to in FL?
PARAPHRASING QUESTIONS ($\mathcal{P}$)
Q_6 What camps are good for a vacation during the summer in FL?
Q_7 What summer camps in FL do you recommend?
USEFUL QUESTIONS ($\mathcal{U}$)
Q_8 Does anyone know a good art summer camp to go to in FL?
Q_9 Are there any good artsy camps for girls in FL?
Q_{10} What are some summer camps for like singing in Florida?
Q_{11} What is a good cooking summer camp in FL?
Q_{12} Do you know of any summer camps in Tampa, FL?
Q_{13} What is a good summer camp in Sarasota FL for a 12 year old?
Q_{14} Can you please help me find a surfing summer camp for beginners in Treasure Coast, FL?
Q_{15} Are there any acting summer camps and/or workshops in the Orlando, FL area?
Q_{16} Does anyone know any volleyball camps in Miramar, FL?
Q_{17} Does anyone know about any cool science camps in Miami?
Q_{18} What's a good summer camp you've ever been to?
NEUTRAL QUESTIONS ($\mathcal{N}$)
Q_{19} What's a good summer camp in Canada?
Q_{20} What's the summer like in Florida?

Table 1: A question group.

and other online repositories that have a high cosine similarity with the reference question. Due to the significant lexical overlap between the questions, this is a rather difficult dataset, especially for ranking methods that rely exclusively on bag-of-words measures. Inside each group, the questions are manually annotated with a partial order relation, according to their utility with respect to the reference question. We shall use the notation $\langle Q_i \succ Q_j | Q_r \rangle$ to encode the fact that question Q_i is *more useful than* question Q_j with respect to the reference question Q_r. Similarly, $\langle Q_i = Q_j \rangle$ will be used to express the fact that questions Q_i and Q_j are reformulations of each other (the reformulation relation is independent of the reference question). The partial ordering among the questions Q_0 to Q_4 above can therefore be expressed concisely as follows: $\langle Q_0 = Q_1 \rangle$, $\langle Q_1 \succ Q_2 | Q_0 \rangle$, $\langle Q_1 \succ Q_3 | Q_0 \rangle$, $\langle Q_2 \succ Q_4 | Q_0 \rangle$, $\langle Q_3 \succ Q_4 | Q_0 \rangle$. Note that we do not explicitly annotate the relation $\langle Q_1 \succ Q_4 | Q_0 \rangle$, since it can be inferred based on the transitivity of the *more useful than* relation: $\langle Q_1 \succ Q_2 | Q_0 \rangle \wedge \langle Q_2 \succ Q_4 | Q_0 \rangle \Rightarrow \langle Q_1 \succ Q_4 | Q_0 \rangle$. Also note that no relation is specified between Q_2 and Q_3, and similarly no relation can be inferred between these two questions. This reflects our belief that, in the absence of any additional information regarding the user or the "turtle" referenced in Q_0, we cannot compare questions Q_2 and Q_3 in terms of their usefulness with respect to Q_0.

Table 1 shows another reference question Q_5 from our dataset, together with its annotated group of questions Q_6 to Q_{20}. In order to make the annotation process easier and reproducible, we divide it into two levels of annotation. During the first annotation stage (L_1), each question group is partitioned manually into 3 subgroups of questions:

- $\mathcal{P}$ is the set of *paraphrasing* questions.
- $\mathcal{U}$ is the set of *useful* questions.
- $\mathcal{N}$ is the set of *neutral* questions.

A question is deemed useful if its expected answer may overlap in information content with the expected answer of the reference question. The expected answer of a neutral question, on the other hand, should be irrelevant with respect to the reference question. Let Q_r be the reference question, $Q_p \in \mathcal{P}$ a paraphrasing question, $Q_u \in \mathcal{U}$ a useful question, and $Q_n \in \mathcal{N}$ a neutral question. Then the following relations are assumed to hold among these questions:

1. $\langle Q_p \succ Q_u | Q_r \rangle$: a *paraphrasing* question is more useful than a *useful* question.

2. $\langle Q_u \succ Q_n | Q_r \rangle$: a *useful* question is more useful than a *neutral* question.

We also assume that, by transitivity, the following ternary relations also hold: $\langle Q_p \succ Q_n | Q_r \rangle$, i.e. a *paraphrasing* question is more useful than a *neutral* question. Furthermore, if $Q_{p_1}, Q_{p_2} \in \mathcal{P}$ are two paraphrasing questions, this implies $\langle Q_{p_1} = Q_{p_2} | Q_r \rangle$.

For the vast majority of questions, the first annotation stage is straightforward and non-controversial. In the second annotation stage (L_2), we perform a finer annotation of relations between questions in the middle group $\mathcal{U}$. Table 1 shows two such relations (using indentation): $\langle Q_8 \succ Q_9 | Q_5 \rangle$ and $\langle Q_8 \succ Q_{10} | Q_5 \rangle$. Question Q_8 would have been a rephrasing of the reference question, were it not for the noun "art" modifying the focus noun phrase "summer camp". Therefore, the information content of the answer to Q_8 is strictly subsumed in the information content associated with the answer to Q_5. Similarly, in Q_9 the focus noun phrase is further specialized through the prepositional phrase "for girls". Therefore, (an answer to) Q_9 is less *useful* to Q_5 than (an answer to) Q_8, i.e. $\langle Q_8 \succ Q_9 | Q_5 \rangle$. Furthermore, the focus "art summer camp" in Q_8 conceptually subsumes the focus "summer camps for singing" in Q_{10}, therefore $\langle Q_8 \succ Q_{10} | Q_5 \rangle$.

Table 2 below presents the following statistics on the annotated dataset: the number of reference questions (Q_r), the total number of paraphrasings ($\mathcal{P}$), the total number of useful questions ($\mathcal{U}$), the total number of neutral questions ($\mathcal{N}$), and the total number of *more useful than* ordered pairs encoded in the dataset, either explicitly or through transitivity, in the two annotation levels L_1 and L_2.

Q_r	$\mathcal{P}$	$\mathcal{U}$	$\mathcal{N}$	L_1	L_2
60	177	847	427	7,378	7,639

Table 2: Dataset statistics.

3 Question Ranking Methods

An ideal question ranking method would take an arbitrary triplet of questions Q_r, Q_i and Q_j as input, and output an ordering between Q_i and Q_j with respect to the reference question Q_r, i.e. one of $\langle Q_i \succ Q_j | Q_r \rangle$, $\langle Q_i = Q_j | Q_r \rangle$, or $\langle Q_j \succ Q_i | Q_r \rangle$. One approach is to design a *usefulness* function $u(Q_i, Q_r)$ that measures how useful question Q_i is for the reference question Q_r, and define the *more useful than* ($\succ$) relation as follows:

$$\langle Q_i \succ Q_j | Q_r \rangle \Leftrightarrow u(Q_i, Q_r) > u(Q_j, Q_r)$$

If we define $I(Q)$ to be the information need associated with question Q, then $u(Q_i, Q_r)$ could be defined as a measure of the relative overlap between $I(Q_i)$ and $I(Q_r)$. Unfortunately, the information need is a concept that, in general, is defined only intensionally and therefore it is difficult to measure. For lack of an operational definition of the information need, we will approximate $u(Q_i, Q_r)$ directly as a measure of the similarity between Q_i and Q_r. The similarity between two questions can be seen as a special case of text-to-text similarity, consequently one possibility is to use a general text-to-text similarity function such as *cosine similarity* in the vector space model (Baeza-Yates and Ribeiro-Neto, 1999):

$$cos(Q_i, Q_r) = \frac{Q_i^T Q_r}{\|Q_i\| \|Q_r\|}$$

Here, Q_i and Q_r denote the corresponding *tf*$\times$*idf* vectors. As a measure of question-to-question similarity, cosine has two major drawbacks:

1. As an exclusively lexical measure, it is oblivious to the meanings of words in each question.

2. Questions are treated as bags-of-words, and thus important structural information is missed.

3.1 Meaning Aware Measures

The three questions below illustrate the first problem associated with cosine similarity. Q_{22} and Q_{23} have the same cosine similarity with Q_{21}, they are therefore indistinguishable in terms of their usefulness to the reference question Q_{21}, even though we expect Q_{22} to be more useful than Q_{23} (a place that sells hydrangea often sells other types of plants too, possibly including cacti).

Q_{21} Where can I buy a hydrangea?

Q_{22} Where can I buy a cactus?

Q_{23} Where can I buy an iPad?

To alleviate the lexical chasm, we can redefine $u(Q_i, Q_r)$ to be the similarity measure proposed by (Mihalcea et al., 2006) as follows:

$$mcs(Q_i, Q_r) \;=\; \frac{\sum_{w \in \{Q_i\}} (maxSim(w, Q_r) * idf(w))}{\sum_{w \in \{Q_i\}} idf(w)} + \frac{\sum_{w \in \{Q_r\}} (maxSim(w, Q_i) * idf(w))}{\sum_{w \in \{Q_r\}} idf(w)}$$

Since scaling factors are immaterial for ranking, we have ignored the normalization constant contained in the original measure. For each word $w \in Q_i$, $maxSim(w, Q_r)$ computes the maximum semantic similarity between w and any word $w_r \in Q_r$. The similarity scores are then weighted by the corresponding *idf*'s, and normalized. A similar score is computed for each word $w \in Q_r$. The score computed by $maxSim$ depends on the actual function used to compute the word-to-word semantic similarity. In this paper, we evaluated four of the knowledge-based measures explored in (Mihalcea et al., 2006): *wup* (Wu and Palmer, 1994), *res* (Resnik, 1995), *lin* (Lin, 1998), and *jcn* (Jiang and Conrath, 1997). Since all these measures are defined on pairs of WordNet concepts, their analogues on word pairs (w_i, w_r) are computed by selecting pairs of WordNet synsets (c_i, c_r) such that w_i belongs to concept c_i, w_r belongs to concept c_r, and (c_i, c_r) maximizes the similarity function. The measure introduced in (Wu and Palmer, 1994) finds the *least common subsumer (LCS)* of the two input concepts in the WordNet hierarchy, and computes the ratio between its depth and the sum of the depths of the two concepts:

$$wup(c_i, c_r) = \frac{2 * depth(lcs(c_i, c_r))}{depth(c_i) + depth(c_r)}$$

Resnik's measure is based on the Information Content (IC) of a concept c defined as the negative log probability $-\log P(c)$ of finding that concept in a large corpus:

$$res(c_i, c_r) = IC(lcs(c_i, c_r))$$

Lin's similarity measure can be seen as a normalized version of Resnik's information content:

$$lin(c_i, c_r) = \frac{2 * IC(lcs(c_i, c_r))}{IC(c_i) + IC(c_r)}$$

Jiang & Conrath's measure is closely related to *lin* and is computed as follows:

$$jcn(c_i, c_r) = [IC(c_i) + IC(c_r) - 2 * IC(lcs(c_i, c_r))]^{-1}$$

3.2 Structure Aware Measures

Cosine similarity, henceforth referred as *cos*, treats questions as bags-of-words. The meta-measure proposed in (Mihalcea et al., 2006), henceforth called *mcs*, treats questions as bags-of-concepts. Consequently, both *cos* and *mcs* may miss important structural information. If we consider the question Q_{24} below as reference, question Q_{26} will be deemed more useful than Q_{25} when using *cos* or *mcs* because of the higher relative lexical and conceptual overlap with Q_{24}. However, this is contrary to the actual ordering $\langle Q_{25} \succ Q_{26} | Q_{24} \rangle$, which reflects that fact that Q_{25}, which expects the same answer type as Q_{24}, should be deemed more useful than Q_{26}, which has a different answer type.

Q_{24} What are some good thriller *movies*?

Q_{25} What are some thriller *movies* with happy ending?

Q_{26} What are some good *songs* from a thriller movie?

The analysis above shows the importance of using the answer type when computing the similarity between two questions. However, instead of relying exclusively on a predefined hierarchy of answer types, we have decided to identify the *question focus* of a question, defined as the set of maximal noun phrases in the question that corefer with the expected answer. Focus nouns such as *movies* and *songs* provide more discriminative information than general answer types such as *products*. We use answer types only for questions such as Q_{27} or Q_{28} below that lack an explicit question focus. In such cases, an artificial question focus is created from the answer type (e.g. *location* for Q_{27}, or *method* for Q_{28}) and added to the set of question words.

Q_{27} *Where* can I buy a good coffee maker?

Q_{28} *How* do I make a pizza?

Let *qsim* be a general bag-of-words question similarity measure (e.g. *cos* or *mcs*). Furthermore, let *wsim* by a generic word meaning similarity measure (e.g. *wup*, *res*, *lin* or *jcn*). The equation below describes a modification of *qsim* that makes it aware of the questions focus:

$$qsim_f(Q_i, Q_r) = wsim(f_i, f_r) * qsim(Q_i - \{f_i\}, Q_r - \{f_r\})$$

Here, Q_i and Q_r refer both to the questions and their sets of words, while f_i and f_r stand for the corresponding focus words. We define $qsim$ to return 1 if one of its arguments is an empty set, i.e. $qsim(\emptyset, \_) = qsim(\_, \emptyset) = 1$. The new similarity measure $qsim_f$ multiplies the semantic similarity between the two focus words with the bag-of-words similarity between the remaining words in the two questions. Consequently, the word "movie" in Q_{26} will not be compared with the word "movies" in Q_{24}, and therefore Q_{26} will receive a lower utility score than Q_{25}.

In addition to the question focus, the *main verb* of a question can also provide key information in estimating question-to-question similarity. We define the main verb to be the content verb that is highest in the dependency tree of the question, e.g. *buy* for Q_{27}, or *make* for Q_{28}. If the question does not contain a content verb, the main verb is defined to be the highest verb in the dependency tree, as for example *are* in Q_{24} to Q_{26}. The utility of a question's main verb in judging its similarity to other questions can be seen more clearly in the questions below, where Q_{29} is the reference:

Q_{29} How can I *transfer* music from iTunes to my iPod?

Q_{30} How can I *upload* music to my iPod?

Q_{31} How can I *play* music in iTunes?

The fact that *upload*, as the main verb of Q_{30}, is more semantically related to *transfer* (*upload* is a hyponym of *transfer* in WordNet) is essential in deciding that $\langle Q_{30} \succ Q_{31} | Q_{29} \rangle$, i.e. Q_{30} is more useful than Q_{31} to Q_{29}.

Like the focus word, the main verb can be incorporated in the question similarity function as follows:

$$qsim_{fv}(Q_i, Q_r) = wsim(f_i, f_r) * wsim(v_i, v_r) * qsim(Q_i - \{f_i, v_i\}, Q_r - \{f_r, v_r\})$$

The new measure $qsim_{fv}$ takes into account both the focus words and the main verbs when estimating the semantic similarity between questions. When decomposing the questions into focus words, main verbs and the remaining words, we have chosen to multiply the corresponding similarities instead of, for example, summing them. Consequently, a close to zero score in each of them would drive the entire similarity to zero. This reflects the belief that question similarity is sensitive to each component of a question.

4 Experimental Evaluation

We use the question ranking dataset described in Section 2 to evaluate the two similarity measures cos and mcs, as well as their structured versions cos_f, cos_{fv}, mcs_f, and mcs_{fv}. We report one set of results for each of the four word similarity measures *wup*, *res*, *lin* or *jcn*. Each question similarity measure is evaluated in terms of its accuracy on the set of ordered pairs for each of the two annotation levels described in Section 2. Thus, for the first annotation level (L_1) , we evaluate only over the set of relations defined across the three

Question similarity ($qsim$)	Word similarity ($wsim$)							
	wup		*res*		*lin*		*jcn*	
	L_1	L_2	L_1	L_2	L_1	L_2	L_1	L_2
cos	69.1	69.3	69.1	69.3	69.1	69.3	69.1	69.3
cos_f	69.9	70.1	**72.5**	**72.7**	71.0	71.2	69.6	69.8
cos_{fv}	69.9	70.1	**72.5**	**72.6**	71.0	71.2	69.6	69.8
mcs	62.6	62.5	65.0	65.0	65.6	65.7	66.8	66.9
mcs_f	64.2	64.4	68.5	68.5	**68.8**	**68.9**	67.2	67.4
mcs_{fv}	65.8	66.0	68.8	68.8	**69.7**	**69.8**	67.7	67.8

Table 3: Accuracy results, with and without meaning and structure information.

sets $\mathcal{R}$, $\mathcal{U}$, and $\mathcal{N}$. If $\langle Q_i \succ Q_j | Q_r \rangle$ is a relation specified in the annotation, we consider the tuple $\langle Q_i, Q_j, Q_r \rangle$ correctly classified if and only if $u(Q_i, Q_r) > u(Q_j, Q_r)$, where u is the question similarity measure (Section 3). For the second annotation level (L_2), we also consider the relations annotated between *useful* questions inside the group $\mathcal{U}$.

We used the NLTK [1] implementation of the four similarity measures *wup*, *res*, *lin* or *jcn*. The *idf* values for each word were computed from frequency counts over the entire Wikipedia. For each question, the *focus* is identified automatically by an SVM tagger trained on a separate corpus of 2,000 questions manually annotated with focus information. The SVM tagger uses a combination of lexico-syntactic features and a quadratic kernel to achieve a 93.5% accuracy in a 10-fold cross validation evaluation on the 2,000 questions. The *main verb* of a question is identified deterministically using a breadth first traversal of the dependency tree.

The overall accuracy results presented in Table 3 show that using the focus word improves the performance across all 8 combinations of question and word similarity measures. For cosine similarity, the best performing system uses the focus words and Resnik's similarity function to obtain a 3.4% increase in accuracy. For the meaning aware similarity *mcs*, the best performing system uses the focus words, the main verb and Lin's word similarity to achieve a 4.1% increase in accuracy. The improvement due to accounting for focus words is consistent, whereas adding the main verb seems to improve the performance only for *mcs*, although not by a large margin. The second level of annotation brings 261 more relations in the dataset, some of them more difficult to annotate when compared with the three groups in the first level. Nevertheless, the performance either remains the same (somewhat expected due to the relatively small number of additional relations), or is marginally better. The random baseline – assigning a random similarity value to each pair of questions – results in 50% accuracy. A somewhat unexpected result is that *mcs* does not perform better than *cos* on this dataset. After analysing the result in more detail, we have noticed that *mcs* seems to be less resilient than *cos* to variations in the length of the questions. The Microsoft paraphrase corpus was specifically designed such that "the length of the shorter of the two sentences, in words, is at least 66% that of the longer" (Dolan and Brockett, 2005), whereas in our dataset the two questions in a pair can have significantly different lengths [2].

The questions in each of the 60 groups have a high degree of lexical overlap, making the dataset especially difficult. In this context, we believe the results are encouraging. We expect to obtain further improvements in accuracy by allowing relations between all the words in a question to influence the overall similarity measure. For example, question Q_{19} has the same focus word as the reference question Q_5 (repeated below), yet the difference between the focus word prepositional modifiers makes it a neutral question.

[1] http://www.nltk.org

[2] Our implementation of *mcs* did performed better than *cos* on the Microsoft dataset.

Q_5 What's a good summer camp to go to in FL?

Q_{19} What's a good summer camp in Canada?

Some of the questions in our dataset illustrate the need to design a word similarity function specifically tailored to reflect how words change the relative usefulness of a question. In the set of questions below, in deciding that Q_{33} and Q_{34} are more useful than Q_{36} for the reference question Q_{32}, an ideal question ranker needs to know that the "Mayflower Hotel" and the "Queensboro Bridge" are in the proximity of "Midtown Manhattan", and that proximity relations are relevant when asking for directions. A coarse measure of proximity can be obtained for the pair ("Manhattan", "Queensboro Bridge") by following the *meronymy* links connecting the two entities in WordNet. However, a different strategy needs to be devised for entities such as "Mayflower Hotel", "JFK", or "La Guardia" which are not covered in WordNet.

Q_{32} What is the best way to get to Midtown Manhattan from JFK?

Q_{33} What's the best way from JFK to Mayflower Hotel?

Q_{34} What's the best way from JFK to Queensboro Bridge?

Q_{35} How do I get from Manhattan to JFK airport by train?

Q_{36} What is the best way to get to LaGuardia from JFK?

Finally, to realize why question Q_{35} is useful one needs to know that, once directions on how to get by train from location X to location Y are known, then normally it suffices to reverse the list of stops in order to obtain directions on how to get from Y back to X.

5 Future Work

We plan to integrate the entire dependency structure of the question in the overall similarity measure, possibly by defining kernels between questions in a maximum margin model for ranking. We also plan to extend the word similarity functions to better reflect the types of relations that are relevant when measuring question utility, such as proximity relations between locations. Furthermore, we intend to take advantage of databases of interrogative paraphrases and paraphrase patterns that were created in previous research on question reformulation.

6 Conclusion

We presented a novel question ranking task in which previously known questions are ordered based on their relative utility with respect to a new, reference question. We created a dataset of 60 groups of questions [3] annotated with a partial order relation reflecting the relative utility of questions inside each group, and used it to evaluate the ranking performance of several meaning and structure aware utility functions. Experimental results demonstrate the importance of using structural information in judging the relative usefulness of questions. We believe that the new perspective on ranking questions has the potential to significantly improve the usability of social QA sites.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful suggestions.

References

Baeza-Yates, Ricardo and Berthier Ribeiro-Neto. 1999. *Modern Information Retrieval*. ACM Press, New York.

Bernhard, Delphine and Iryna Gurevych. 2008. Answering learners' questions by retrieving question paraphrases from social Q&A sites. In *EANL '08: Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, pages 44–52, Morristown, NJ, USA. Association for Computational Linguistics.

Dolan, William B. and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, pages 9–16.

[3]The dataset will be made publicly available.

Hermjakob, Ulf, Abdessamad Echihabi, and Daniel Marcu. 2002. Natural language based reformulation resource and web exploitation for question answering. In *Proceedings of TREC-2002*.

Jeon, Jiwoon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM'05)*, pages 84–90, New York, NY, USA. ACM.

Jiang, J.J. and D.W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *Proceedings of the International Conference on Research in Computational Linguistics*, pages 19–33.

Jijkoun, Valentin and Maarten de Rijke. 2005. Retrieving answers from frequently asked questions pages on the Web. In *Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM'05)*, pages 76–83, New York, NY, USA. ACM.

Lin, Dekang. 1998. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning (ICML '98)*, pages 296–304, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Mihalcea, Rada, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st national conference on Artificial intelligence (AAAI'06)*, pages 775–780. AAAI Press.

Resnik, Philip. 1995. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI'95: Proceedings of the 14th international joint conference on Artificial intelligence*, pages 448–453, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Tomuro, Noriko. 2003. Interrogative reformulation patterns and acquisition of question paraphrases. In *Proceedings of the Second International Workshop on Paraphrasing*, pages 33–40, Morristown, NJ, USA. Association for Computational Linguistics.

Wu, Zhibiao and Martha Palmer. 1994. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, Morristown, NJ, USA. Association for Computational Linguistics.

Zhao, Shiqi, Ming Zhou, and Ting Liu. 2007. Learning question paraphrases for QA from Encarta logs. In *Proceedings of the 20th international joint conference on Artifical intelligence (IJCAI'07)*, pages 1795–1800, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Simultaneous Ranking and Clustering of Sentences: A Reinforcement Approach to Multi-Document Summarization

[1]Xiaoyan Cai, [1]Wenjie Li, [1]You Ouyang, [2]Hong Yan

[1]Department of Computing, The Hong Kong Polytechnic University
{csxcai,cswjli,csyouyang}@comp.polyu.edu.hk

[2]Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University
lgthyan@polyu.edu.hk

Abstract

Multi-document summarization aims to produce a concise summary that contains salient information from a set of source documents. In this field, sentence ranking has hitherto been the issue of most concern. Since documents often cover a number of topic themes with each theme represented by a cluster of highly related sentences, sentence clustering was recently explored in the literature in order to provide more informative summaries. Existing cluster-based ranking approaches applied clustering and ranking in isolation. As a result, the ranking performance will be inevitably influenced by the clustering result. In this paper, we propose a reinforcement approach that tightly integrates ranking and clustering by mutually and simultaneously updating each other so that the performance of both can be improved. Experimental results on the DUC datasets demonstrate its effectiveness and robustness.

1 Introduction

Automatic multi-document summarization has drawn increasing attention in the past with the rapid growth of the Internet and information explosion. It aims to condense the original text into its essential content and to assist in filtering and selection of necessary information. So far extractive summarization that directly extracts sentences from documents to compose summaries is still the mainstream in this field. Under this framework, sentence ranking is the issue of most concern.

Though traditional feature-based ranking approaches and graph-based approaches employed quite different techniques to rank sentences, they have at least one point in common, i.e., all of them focused on sentences only, but ignored the information beyond the sentence level (referring to Figure 1(a)). Actually, in a given document set, there usually exist a number of themes (or topics) with each theme represented by a cluster of highly related sentences (Harabagiu and Lacatusu, 2005; Hardy et al., 2002). These theme clusters are of different size and especially different importance to assist users in understanding the content in the whole document set. The cluster level information is supposed to have foreseeable influence on sentence ranking.

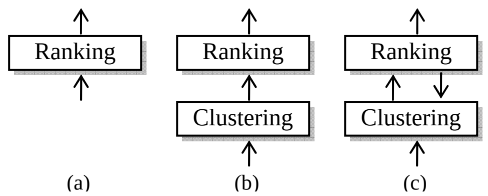


Figure 1. Ranking vs. Clustering

In order to enhance the performance of summarization, recently cluster-based ranking approaches were explored in the literature (Wan and Yang, 2006; Sun et al, 2007; Wang et al, 2008a,b; Qazvinian and Radev, 2008). Normally these approaches applied a clustering algorithm to obtain the theme clusters first and then ranked the sentences within each cluster or by exploring the interaction between sentences and obtained clusters (referring to Figure 1(b)). In other words, clustering and ranking are regarded as two independent processes in these approaches although the cluster-level information has been incorporated into the sentence ranking process. As a result,

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 134–142,
Beijing, August 2010

the ranking performance is inevitably influenced by the clustering result.

To help alleviate this problem, we argue in this paper that the quality of ranking and clustering can be both improved when the two processes are mutually enhanced (referring to Figure 1(c)). Based on it, we propose a reinforcement approach that updates ranking and clustering interactively and iteratively to multi-document summarization. The main contributions of the paper are three-fold: (1) Three different ranking functions are defined in a bi-type document graph constructed from the given document set, namely global, within-cluster and conditional rankings, respectively. (2) A reinforcement approach is proposed to tightly integrate ranking and clustering of sentences by exploring term rank distributions over the clusters. (3) Thorough experimental studies are conducted to verify the effectiveness and robustness of the proposed approach.

The rest of this paper is organized as follows. Section 2 reviews related work in cluster-based ranking. Section 3 defines ranking functions and explains reinforced ranking and clustering process and its application in multi-document summarization. Section 4 presents experiments and evaluations. Section 5 concludes the paper.

2 Related Work

Clustering has become an increasingly important topic with the explosion of information available via the Internet. It is an important tool in text mining and knowledge discovery. Its ability to automatically group similar textual objects together enables one to discover hidden similarity and key concepts, as well as to summarize a large amount of text into a small number of groups (Karypis et al., 2000).

To summarize a scientific paper, Qazvinian and Radev (2008) presented two sentence selection strategies based on the clusters which were generated by a hierarchical agglomeration algorithm applied in the citation summary network. One was called C-RR, which started with the largest cluster and extracted the first sentence from each cluster in the order they appeared until the summary length limit was reached. The other was called C-LexRank, which was similar to C-RR but adopted LexRank to rank the sentences within each cluster and chose the most salient one.

Meanwhile, Wan and Yang (2008) proposed two models to incorporate the cluster-level information into the process of sentence ranking for generic summarization. While the Cluster-based Conditional Markov Random Walk model (ClusterCMRW) incorporated the cluster-level information into the text graph and manipulated clusters and sentences equally, the Cluster-based HITS model (ClusterHITS) treated clusters and sentences as hubs and authorities in the HITS algorithm.

Besides, Wang et al. (2008) proposed a language model to simultaneously cluster and summarize documents. Nonnegative factorization was performed on the term-document matrix using the term-sentence matrix as the base so that the document-topic and sentence-topic matrices could be constructed, from which the document clusters and the corresponding summary sentences were generated simultaneously.

3 A Reinforcement Approach to Multi-document Summarization

3.1 Document Bi-type Graph

First of all, let's introduce the sentence-term bi-type graph model for a set of given documents D, based on which the algorithm of reinforced ranking and clustering is developed. Let $G = <V, E, W>$, where V is the set of vertices that consists of the sentence set $S = \{s_1, s_2, \dots, s_n\}$ and the term set $T = \{t_1, t_2, \dots, t_m\}$, i.e., $V = S \cup T$, E is the set of edges that connect the vertices, i.e., $E = \{<v_i, v_j> | v_i, v_j \in V\}$. W is the adjacency matrix in which the element w_{ij} represents the weight of the edge connecting v_i and v_j. Formally, W can be decomposed into four blocks, i.e., W_{SS}, W_{ST}, W_{TS} and W_{TT}, each representing a sub-graph of the textual objects indicated by the subscripts. W can be written as

$$W = \begin{pmatrix} W_{SS} & W_{ST} \\ W_{TS} & W_{TT} \end{pmatrix},$$

where $W_{ST}(i, j)$ is the number of times the term t_j appears in the sentence s_i. $W_{SS}(i,j)$ is the number of common terms in the sentences s_i and s_j. W_{TS} is equal to W_{ST}^T as the relationships between terms and sentences are symmetric. For simplification, in this study we assume there is no direct relationships between terms, i.e., $W_{TT} = 0$. In the future, we will explore effective ways to integrate term semantic relationships into the model.

3.2 Basic Ranking Functions

Recall that our ultimate goal is sentence ranking. As an indispensable part of the approach, the basic ranking functions need to be defined first.

3.2.1 Global Ranking (without Clustering)

Let $r(s_i)$ (i=1, 2, …, n) and $r(t_j)$ (j=1, 2, …, m) denote the ranking scores of the sentence s_i and the term t_j in the whole document set, respectively. Based on the assumptions that

> *"Highly ranked terms appear in highly ranked sentences, while highly ranked sentences contain highly ranked terms. Moreover, a sentence is ranked higher if it contains many terms that appear in many other highly ranked sentences."*

we define

$$r(s_i) = \lambda \cdot \sum_{j=1}^{m} W_{ST}(i,j) \cdot r(t_j) + (1-\lambda) \cdot \sum_{j=1}^{n} W_{SS}(i,j) \cdot r(s_j) \quad (1)$$

and

$$r(t_j) = \sum_{i=1}^{n} W_{TS}(j,i) \cdot r(s_i) \,. \quad (2)$$

For calculation purpose, $r(s_i)$ and $r(t_j)$ are normalized by

$$r(s_i) \leftarrow \frac{r(s_i)}{\sum_{i'=1}^{n} r(s_{i'})} \text{ and } r(t_j) \leftarrow \frac{r(t_j)}{\sum_{j'=1}^{m} r(t_{j'})} \,.$$

Equations (1) and (2) can be rewritten using the matrix form, i.e.,

$$\begin{cases} r(S) = \lambda \cdot \dfrac{W_{ST} \cdot r(T)}{\| W_{ST} \cdot r(T) \|} + (1-\lambda) \cdot \dfrac{W_{SS} \cdot r(S)}{\| W_{SS} \cdot r(S) \|} \\ r(T) = \dfrac{W_{TS} \cdot r(S)}{\| W_{TS} \cdot r(S) \|} \end{cases} . \quad (3)$$

We call $r(S)$ and $r(T)$ the "**global ranking functions**", because at this moment sentence clustering is not yet involved and all the sentences/terms in the whole document set are ranked together.

Theorem: The solution to $r(S)$ and $r(T)$ given by Equation (3) is the primary eigenvector of $\lambda \cdot W_{ST} \cdot W_{TS} + (1-\lambda) \cdot W_{SS}$ and $\lambda \cdot W_{TS}(I-(1-\lambda) \cdot W_{SS})^{-1} \cdot W_{ST}$, respectively.

Proof: Combine Equations (1) and (2), we get

$$r(S) = \lambda \cdot \frac{W_{ST} \cdot \dfrac{W_{TS} \cdot r(S)}{\| W_{TS} \cdot r(S) \|}}{\left\| W_{ST} \cdot \dfrac{W_{TS} \cdot r(S)}{\| W_{TS} \cdot r(S) \|} \right\|} + (1-\lambda) \cdot \frac{W_{SS} \cdot r(S)}{\| W_{SS} \cdot r(S) \|}$$

$$= \lambda \cdot \frac{W_{ST} \cdot W_{TS} \cdot r(S)}{\| W_{ST} \cdot W_{TS} \cdot r(S) \|} + (1-\lambda) \cdot \frac{W_{SS} \cdot r(S)}{\| W_{SS} \cdot r(S) \|}$$

As the iterative process is a power method, it is guaranteed that $r(S)$ converges to the primary eigenvector of $\lambda \cdot W_{ST} \cdot W_{TS} + (1-\lambda) \cdot W_{SS}$. Similarly, $r(T)$ is guaranteed to converge to the primary eigenvector of $\lambda \cdot W_{TS}(I-(1-\lambda) \cdot W_{SS})^{-1} \cdot W_{ST}$. ■

3.2.2 Local Ranking (within Clusters)

Assume now K theme clusters have been generated by certain clustering algorithm, denoted as $C = \{C_1, C_2, \ldots, C_K\}$ where C_k (k=1, 2, …, K) represents a cluster of highly related sentences S_{C_k} ($\in C_k$) which contain the terms T_{C_k} ($\in C_k$). The sentences and terms within the cluster C_k form a cluster bi-type graph with the adjacency matrix W_{C_k}. Let $r_{C_k}(S_{C_k})$ and $r_{C_k}(T_{C_k})$ denote the ranking scores of S_{C_k} and T_{C_k} within C_k. They are calculated by an equation similar to Equation (3) by replacing the document level adjacency matrix W with the cluster level adjacency matrix W_{C_k}. We call $r_{C_k}(S_{C_k})$ and $r_{C_k}(T_{C_k})$ the "**within-cluster ranking functions**" with respect to the cluster C_k. They are the local ranking functions, in contrast to $r(S)$ and $r(T)$ that rank all the sentences and terms in the whole document set D. We believe that it will benefit sentence overall ranking when knowing more details about the ranking results at the finer granularity of theme clusters, instead of at the coarse granularity of the whole document set.

3.2.3 Conditional Ranking (*across Clusters*)

To facilitate the discovery of rank distributions of terms and sentences over all the theme clusters, we further define two "**conditional ranking functions**" $r(S \mid C_k)$ and $r(T \mid C_k)$. These rank distributions are necessary for the parameter estimation during the reinforcement process introduced later. The conditional ranking score of the term t_j on the cluster C_k, i.e., $r(T \mid C_k)$ is directly derived from T_{C_k}, i.e., $r(t_j \mid C_k) = r_{C_k}(t_j)$ if $t_j \in C_k$, and $r(t_j \mid C_k) = 0$ otherwise. It is further normalized as

$$r(t_j \mid C_k) = \frac{r(t_j \mid C_k)}{\sum_{j=1}^{m} r(t_j \mid C_k)}. \quad (4)$$

Then the conditional ranking score of the sentence s_i on the cluster C_k is deduced from the terms that are included in s_i, i.e.,

$$r(s_i \mid C_k) = \frac{\sum_{j=1}^{m} W_{ST}(i,j) \cdot r(t_j \mid C_k)}{\sum_{i=1}^{n} \sum_{j=1}^{m} W_{ST}(i,j) \cdot r(t_j \mid C_k)}. \quad (5)$$

Equation (5) can be interpreted as that the conditional rank of s_i on C_k is higher if many terms in s_i are ranked higher in C_k. Now we have sentence and term conditional ranks over all the theme clusters and are ready to introduce the reinforcement process.

3.3 Reinforcement between Within-Cluster Ranking and Clustering

The conditional ranks of the term t_j across the K theme clusters can be viewed as a rank distribution. Then the rank distribution of the sentence s_i can be considered as a mixture model over K conditional rank distributions of the terms contained in the sentence s_i. And the sentence s_i can be represented as a K-dimensional vector in the new measure space, in which the vectors can be used to guide the sentence clustering update. Next, we will explain the mixture model of sentence and use EM algorithm (Bilmes, 1997) to get the component coefficients of the model. Then, we will present the similarity measure between sentence and cluster, which is used to adjust the clusters that the sentences belong to and in turn modify within-cluster ranking for the sentences in the updated clusters.

3.3.1 Sentence Mixture Model

For each sentence s_i, we assume that it follows the distribution $r(T \mid s_i)$ to generate the relationship between the sentence s_i and the term set T. This distribution can be considered as a mixture model over K component distributions, i.e. the term conditional rank distributions across K theme clusters. We use $\gamma_{\mathrm{i},k}$ to denote the probability that s_i belongs to C_k, then $r(T \mid s_i)$ can be modeled as:

$$r(T \mid s_i) = \sum_{k=1}^{K} \gamma_{\mathrm{i},k} \cdot r(T \mid C_k) \text{ and } \sum_{k=1}^{K} \gamma_{\mathrm{i},k} = 1. \quad (6)$$

$\gamma_{i,k}$ can be explained as $p(C_k \mid s_i)$ and calculated by the Bayesian equation $p(C_k \mid s_i) \propto p(s_i \mid C_k) \cdot p(C_k)$, where $p(s_i \mid C_k)$ is assumed to be $r(s_i \mid C_k)$ obtained from the conditional rank of s_i on C_k as introduced before and $p(C_k)$ is the prior probability.

3.3.2 Parameter Estimation

We use EM algorithm to estimate the component coefficients $\gamma_{\mathrm{i},k}$ along with $\{p(C_k)\}$. A hidden variable C_z, $z \in \{1,2,\ldots,K\}$ is used to denote the cluster label that a sentence term pair (s_i, t_j) are from. In addition, we make the independent assumption that the probability of s_i belonging to C_k and the probability of t_j belonging to C_k are independent, i.e., $p(s_i, t_j \mid C_k) = p(s_i \mid C_k) \cdot p(t_j \mid C_k)$, where $p(s_i, t_j \mid C_k)$ is the probability of s_i and t_j both belonging to C_k. Similarly, $p(t_j \mid C_k)$ is assumed to be $r(t_j \mid C_k)$.

Let Θ be the parameter matrix, which is a $n \times K$ matrix $\Theta_{n \times K} = \{\gamma_{i,k}\}$ $(i = 1,\ldots,n;$ $k = 1,\ldots,K)$. The best Θ is estimated from the relationships observed in the document bi-type graph, i.e., W_{ST} and W_{SS}. The likelihood of generating all the relationships under the parameter Θ can be calculated as:

$$L'(\Theta \mid W_{ST}, W_{SS}) = p(W_{ST} \mid \Theta) \cdot p(W_{SS} \mid \Theta)$$
$$= \prod_{i=1}^{n} \prod_{j=1}^{m} p(s_i, t_j \mid \Theta)^{W_{ST}(i,j)} \cdot \prod_{i=1}^{n} \prod_{j=1}^{n} p(s_i, s_j \mid \Theta)^{W_{SS}(i,j)}$$

where $p(s_i, t_j | \Theta)$ is the probability that s_i and t_j both belong to the same cluster, given the current parameter. As $p(s_i, s_j | \Theta)$ does not contain variables from Θ, we only need to consider maximizing the first part of the likelihood in order to get the best estimation of Θ. Let $L(\Theta | W_{ST})$ be the first part of likelihood.

Taking into account the hidden variable C_z, the complete log-likelihood can be written as

$$\log L(\Theta | W_{ST}, C_Z) = \log \prod_{i=1}^{n} \prod_{j=1}^{m} \left(p(s_i, t_j, C_z | \Theta)\right)^{W_{ST}(i,j)}$$
$$= \log \prod_{i=1}^{n} \prod_{j=1}^{m} \left(p(s_i, t_j | C_z, \Theta) \cdot p(C_z | \Theta)\right)^{W_{ST}(i,j)}$$
$$= \sum_{i=1}^{n} \sum_{j=1}^{m} W_{ST}(i,j) \cdot \log\left(p_Z(s_i, t_j) \cdot p(C_z | \Theta)\right).$$

In the E-step, given the initial parameter Θ^0, which is set to $\gamma_{i,k}^0 = 1/K$ for all *i* and *k*, the expectation of log-likelihood under the current distribution of C_Z is:

$$Q(\Theta, \Theta^0) = E_{f(C_Z | W_{ST}, \Theta^0)}(\log L(\Theta | W_{ST}, C_Z)$$
$$= \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j=1}^{m} W_{ST}(i,j) \cdot \log(p_k(s_i, t_j)) \cdot p(C_z = C_k | s_i, t_j, \Theta^0) +$$
$$\sum_{i=1}^{n} \sum_{k=1}^{K} \sum_{j=1}^{m} W_{ST}(i,j) \cdot \log(p(C_z = C_k | \Theta)) \cdot p(C_z = C_k | s_i, t_j, \Theta^0)$$

The conditional distribution in the above equation, i.e., $p(C_z = C_k | s_i, t_j, \Theta^0)$, can be calculated using the Bayesian rule as follows:

$$\begin{aligned} & p(C_z = C_k | s_i, t_j, \Theta^0) \\ & \propto p(s_i, t_j | C_z = C_k, \Theta^0) p(C_z = C_k | \Theta^0). \qquad (7) \\ & \propto p^0(s_i | C_k) p^0(t_j | C_k) p^0(C_z = C_k) \end{aligned}$$

In the M-Step, we first get the estimation of $p(C_z = C_k)$ by maximizing the expectation $Q(\Theta, \Theta^0)$. By introducing a Lagrange multiplier λ, we get the equation below.

$$\frac{\partial}{\partial p(C_z = C_k)} [Q(\Theta, \Theta^0) + \lambda(\sum_{k=1}^{K} p(C_z = C_k) - 1)] = 0 \Rightarrow$$
$$\sum_{i=1}^{n} \sum_{j=1}^{m} W_{ST}(i,j) \frac{1}{p(C_z = C_k)} p(C_z = C_k | s_i, t_j, \Theta^0) + \lambda = 0$$

Thus, the estimation of $p(C_z = C_k)$ given previous Θ^0 is

$$p(C_z = C_k) = \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} W_{ST}(i,j) p(C_z = C_k | s_i, t_j, \Theta^0)}{\sum_{i=1}^{n} \sum_{j=1}^{m} W_{ST}(i,j)}. \quad (8)$$

Then, the parameters $\gamma_{i,k}$ can be calculated with the Bayesian rule as

$$\gamma_{i,k} = \frac{p(s_i | C_k) p(C_z = C_k)}{\sum_{l=1}^{K} p(s_i | C_l) p(C_z = C_l)}. \quad (9)$$

By setting $\Theta^0 = \Theta$, the whole process can be repeated. The updating rules provided in Equations (7)-(9) are applied at each iteration. Finally Θ will converge to a local maximum. A similar estimation process has been adopted in (Sun et al., 2009), which was used to estimate the component coefficients for author-conference networks.

3.3.3 *Similarity Measure*

After we get the estimations of the component coefficients $\gamma_{i,k}$ for s_i, s_i will be represented as a K dimensional vector $\vec{s_i} = (\gamma_{i,1}, \gamma_{i,2}, \ldots, \gamma_{i,K})$. The center of each cluster can thus be calculated accordingly, which is the mean of $\vec{s_i}$ for all s_i in the same cluster, i.e.,

$$\overrightarrow{Center}_{C_k} = \frac{\sum_{s_i \in C_k} \vec{s_i}}{|C_k|},$$

where $|C_k|$ is the size of C_k.

Then the similarity between each sentence and each cluster can be calculated as the cosine similarity between them, i.e.,

$$sim(s_i, C_k) = \frac{\sum_{l=1}^{K} \vec{s_i}(l) \overrightarrow{Center}_{C_k}(l)}{\sqrt{\sum_{l=1}^{K} \vec{s_i}(l))^2} \sqrt{\sum_{l=1}^{K} \overrightarrow{Center}_{C_k}(l))^2}}. \quad (10)$$

Finally, each sentence is re-assigned to a cluster that is the most similar to the sentence. Based on the updated clusters, within-cluster ranking is updated accordingly, which triggers the next round of clustering refinement. It is expected that the quality of clusters should be improved during this iterative update process since the similar sentences under new attributes will be grouped together, and meanwhile the quality of ranking will be improved along with the better clusters and thus offers better attributes for further clustering.

3.4 Ensemble Ranking

The overall sentence ranking function f is defined as the ensemble of all the sentence conditional ranking scores on the K clusters.

$$f(s_i) = \sum_{k=1}^{K} \alpha_k \cdot r(s_i \mid C_k), \tag{11}$$

where α_k is a coefficient evaluating the importance of C_k. It can be formulated as the normalized cosine similarity between a theme cluster and the whole document set for generic summarization, or between a theme cluster and a given query for query-based summarization. $\alpha_k \in [0,1]$ and $\sum_{k=1}^{K} \alpha_k = 1$.

Figure 2 below summarizes the whole process that determines the overall sentence ensemble ranking scores.

Input: The bi-type document graph $G = < S \cup T, E, W >$, ranking functions, the cluster number K, $\varepsilon = 1$, $Tre = 0.001$, $IterNum = 10$.
Output: sentence final ensemble ranking vector $f(S)$.

$t \leftarrow 0$;
Get the initial partition for S, i.e. C_k^t, $k = 1,2,\ldots K$, calculate cluster centers $\overrightarrow{Center}_{C_k^t}$ accordingly.
For (t=1; t<IterNum && $\varepsilon > Tre$; t++)
  Calculate the within-cluster ranking $r_{C_k}(T_{C_k})$, $r_{C_k}(S_{C_k})$ and the conditional ranking $r(s_i \mid C_k)$;
  Get new attribute $\vec{s}_i$ for each sentence s_i, and new attribute $\overrightarrow{Center}_{C_k^t}$ for each cluster C_k^t;
  For each sentence s_i in S
    For k=1 to K
      Calculate similarity value $sim(s_i, C_k^t)$
    End For
    Assign s_i to $C_{k_0}^{t+1}$, $k_0 = \arg\max_k sim(s_i, C_k^t)$
  End For
  $\varepsilon = \max_k |\overrightarrow{Center}_{C_k^{t+1}} - \overrightarrow{Center}_{C_k^t}|$
  $t \leftarrow t+1$
End For
For each sentence s_i in S
  For k=1 to K
    $f(s_i) = \sum_{k=1}^{K} \alpha_k \cdot r(s_i \mid C_k)$
  End For
End For

Figure 2. The Overall Sentence Ranking Algorithm

3.5 Summary Generation

In multi-document summarization, the number of documents to be summarized can be very large. This makes information redundancy appears to be more serious in multi-document summarization than in single-document summarization. Redundancy control is necessary. We apply a simple yet effective way to choose summary sentences. Each time, we compare the current candidate sentence to the sentences already included in the summary. Only the sentence that is not too similar to any sentence in the summary (i.e., the cosine similarity between them is lower than a threshold) is selected into the summary. The iteration is repeated until the length of the sentences in the summary reaches the length limitation. In this paper, the threshold is set to 0.7 as always in our past work.

4 Experiments and Evaluations

We conduct the experiments on the DUC 2004 generic multi-document summarization dataset and the DUC 2006 query-based multi-document summarization dataset. According to task definitions, systems are required to produce a concise summary for each document set (without or with a given query description) and the length of summaries is limited to 665 bytes in DUC 2004 and 250 words in DUC 2006.

A well-recognized automatic evaluation toolkit ROUGE (Lin and Hovy, 2003) is used in evaluation. It measures summary quality by counting overlapping units between system-generated summaries and human-written reference summaries. We report two common ROUGE scores in this paper, namely ROUGE-1 and ROUGE-2, which base on Uni-gram match and Bi-gram match, respectively. Documents and queries are pre-processed by segmenting sentences and splitting words. Stop words are removed and the remaining words are stemmed using Porter stemmer.

4.1 Evaluation of Performance

In order to evaluate the performance of reinforced clustering and ranking approach, we compare it with the other three ranking approaches: (1) Global-Rank, which does not apply clustering and simply relies on the sentence global ranking scores to select summary sentences; (2) Local-Rank, which clusters sentences first and then rank sentences within each cluster. A summary is generated in the same way as presented in (Qazvinian and Radev, 2008). The clusters are ordered by decreasing size; (3) Cluster-HITS, which also clusters sentences first, but then regards clusters as hubs and sentences as authorities in the HITS algorithm and uses the obtained authority scores to rank and select sentences. The classical clustering algorithm K-means is used where necessary. For query-based summarization, the additional query-relevance (i.e. the cosine similarity between sentences and query) is involved to re-rank the candidate sentences chosen by the ranking approaches for generic summarization.

Note that K-means requires a predefined cluster number *K*. To avoid exhaustive search for a proper cluster number for each document set, we employ the spectra approach introduced in (Li et al., 2007) to predict the number of the expected clusters. Based on the sentence similarity matrix using the normalized 1-norm, for its eigenvalues λ_i (i=1,2, …, n), the ratio $\alpha_i = \lambda_{i+1} / \lambda_2 (\lambda \geq 1)$ is defined. If $\alpha_i - \alpha_{i+1} > 0.05$ and α_i is still close to 1, then set $K=i+1$. Tables 1 and 2 below compare the performance of the four approaches on DUC 2004 and 2006 according to the calculated *K*.

DUC 2004	ROUGE-1	ROUGE-2
Reinforced	0.37082	0.08351
Cluster-HITS	0.36463	0.07632
Local-Rank	0.36294	0.07351
Global-Rank	0.35729	0.06893

Table 1. Results on the DUC 2004 dataset

DUC 2006	ROUGE-1	ROUGE-2
Reinforced	0.39531	0.08957
Cluster-HITS	0.38315	0.08632
Local-Rank	0.38104	0.08841
Global-Rank	0.37478	0.08531

Table 2. Results on the DUC 2006 dataset

It is not surprised to find that "Global-Rank" shows the poorest performance, when it utilizes the sentence level information only whereas the other three approaches all integrate the additional cluster level information in various ways. In addition, as results illustrate, the performance of "Cluster-HITS" is better than the performance of "Local-Rank". This can be mainly credited to the ability of "Cluster-HITS" to consider not only the cluster-level information, but also the sentence-to-cluster relationships, which are ignored in "Local-Rank". It is happy to see that the proposed reinforcement approach, which simultaneously updates clustering and ranking of sentences, consistently outperforms the other three approaches.

4.2 Analysis of Cluster Quality

Our original intention to propose the reinforcement approach is to hope to generate more accurate clusters and ranking results by mutually refining within-cluster ranking and clustering. In order to check and monitor the variation trend of the cluster quality during the iterations, we define the following measure

$$quan = \sum_{k=1}^{K} \left(\frac{\min_{s_i \in C_k} sim(s_i, C_k)}{\sum_{l=1, l \neq k}^{K} \min_{s_i \in C_k, s_j \in C_l} sim(s_i, s_j)} \right), \quad (12)$$

where $\min_{s_i \in C_k} sim(s_i, C_k)$ denotes the distance between the cluster center and the border sentence in a cluster that is the farthest away from the center. The larger it is, the more compact the cluster is. $\min_{s_i \in C_k, s_j \in C_l} sim(s_i, s_j)$, on the other hand, denotes the distance between the most distant pair of sentences, one from each cluster. The smaller it is, the more separated the two clusters are. The distance is measured by cosine similarity. As a whole, the larger *quan* means the better cluster quality. Figure 3 below plots the values of *quan* in each iteration on the DUC 2004 and 2006 datasets. Note that the algorithm converges in less than 6 rounds and 5 rounds on the DUC 2004 and 2006 datasets, respectively. The curves clearly show the increasment of *quan* and thus the improved cluster quality.

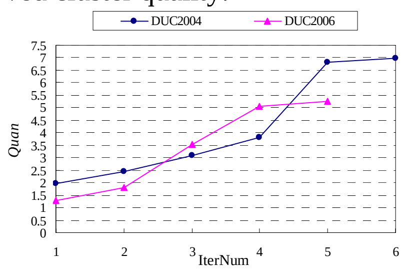


Figure 3. Cluster Quality on DUC 2004 and 2006

While *quan* directly evaluate the quality of the generated clusters, we are also quite interested in whether the improved clusters quality can further enhance the quality of sentence ranking and thus consequently raise the performance of summarization. Therefore, we evaluate the ROUGEs in each iteration as well. Figure 4 below illustrates the changes of ROUGE-1 and ROUGE-2 result on the DUC 2004 and 2006 datasets, respectively. Now, we have come to the positive conclusion.

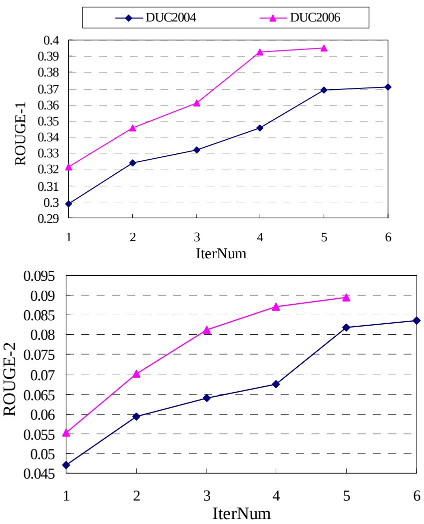


Figure 4. ROUGEs on DUC 2004 and 2006

4.3 Impact of Cluster Numbers

In previous experiments, the cluster number is predicted through the eigenvalues of 1-norm normalized sentence similarity matrix. This number is just the estimated number. The actual number is hard to predict accurately. To further examine how the cluster number influences summarization, we conduct the following additional experiments by varying the cluster number. Given a document set, we let S denote the sentence set in the document set, and set K in the following way:

$$K = \varepsilon \times |S|, \tag{13}$$

where $\varepsilon \in (0,1)$ is a ratio controlling the expected cluster number. The larger ε is, the more clusters will be produced. ε ranges from 0.1 to 0.9 in the experiments. Due to page limitation, we only provide the ROUGE-1 and ROUGE-2 results of the proposed approach, "Cluster-HITS" and "Local-Rank" on the DUC 2004 dataset in Figure 5. The similar curves are also observed on the 2006 dataset.

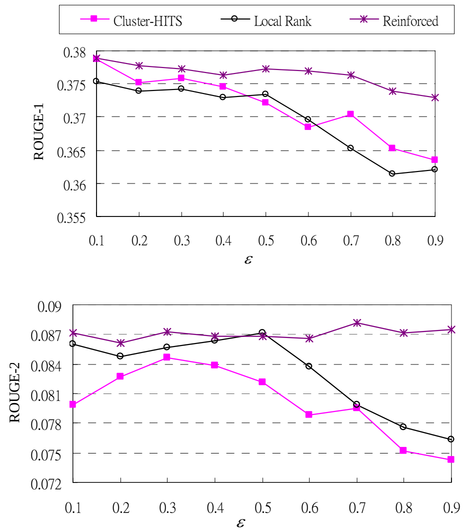


Figure 5. ROUGEs vs. ε on DUC 2004

It is shown that (1) the proposed approach outperforms "Cluster-HITS" and "Local-Rank" in almost all the cases no matter how the cluster number is set; (2) the performances of "Cluster-HITS" and "Local-Rank" are more sensitive to the cluster number and a large number of clusters appears to deteriorate the performances of both. This is reasonable. Actually when ε getting close to 1, "Local-Rank" approaches to "Global-Rank". These results demonstrate the robustness of the proposed approach.

5 Conclusion

In this paper, we present a reinforcement approach that tightly integrates ranking and clustering together by mutually and simultaneously updating each other. Experimental results demonstrate the effectiveness and the robustness of the proposed approach. In the future, we will explore how to integrate term semantic relationships to further improve the performance of summarization.

Acknowledgement

The work described in this paper was supported by an internal grant from the Hong Kong Polytechnic University (G-YG80).

References

J. Bilmes. 1997. *A Gentle Tutorial on the em Algorithm and Its Application to Parameter Wstimation for Gaussian Mixture and Hidden Markov Models*. Technical Report ICSI-TR-97-02, University of Berkeley.

Brin, S., and Page, L. 1998. *The Anatomy of a Large-scale Hypertextual Web Search Engine*. In Proceedings of WWW1998..

Harabagiu S. and Lacatusu F. 2005. *Topic Themes for Multi-Document Summarization*. In Proceedings of SIGIR2005.

Hardy H., Shimizu N., Strzalkowski T., Ting L., Wise G. B., and Zhang X. 2002. *Cross-Document Summarization by Concept Classification*. In Proceedings of SIGIR2002.

Jon M. Kleinberg. 1999. *Authoritative Sources in a Hyperlinked Environment*. In Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms.

Karypis, George, Vipin Kumar and Michael Steinbach. 2000. *A Comparison of Document Clustering Techniques*. KDD workshop on Text Mining.

Lin, C. Y. and Hovy, E. 2000. *The Automated Acquisition of Topic Signature for Text Summarization*. In Proceedings of COLING2000.

Li W.Y., Ng W.K., Liu Y. and Ong K.L. 2007. *Enhancing the Effectiveness of Clustering with Spectra Analysis*. IEEE Transactions on Knowledge and Data Engineering (TKDE). 19(7): 887-902.

Li, F., Tang, Y., Huang, M., Zhu, X. 2009. *Answering Opinion Questions with Random Walks on Graphs*. In Proceedings of ACL2009.

Otterbacher J., Erkan G. and Radev D. 2005. *Using RandomWalks for Question-focused Sentence Retrieval*. In Proceedings of HLT/EMNLP 2005.

Qazvinian V. and Radev D. R. 2008. *Scientific paper summarization using citation summary networks*. In Proceedings of COLING2008.

Sun P., Lee J.H., Kim D.H., and Ahn C.M. 2007. *Multi-Document Using Weighted Similarity Between Topic and Clustering-Based Non-negative Semantic Feature*. APWeb/WAIM 2007.

Sun Y., Han J., Zhao P., Yin Z., Cheng H., and Wu T. 2009. *Rankclus: Integrating Clustering with Ranking for Heterogenous Information Network Analysis*. In Proceedings of EDBT 2009.

Wang D.D., Li T., Zhu S.H., Ding Chris. 2008a *Multi-Document Summarization via Sentence-Level Semantic Analysis and Symmetric Matrix Factorization*. In Proceedings of SIGIR2008.

Wang D.D., Zhu S.H., Li T., Chi Y., and Gong Y.H. 2008b. *Integrating Clustering and Multi-Document Summarization to Improve Document Understanding*. In Proceedings of CIKM 2008.

Wan X. and Yang J. 2006. *Improved Affinity Graph based Multi-Document Summarization*. In Proceedings of HLT-NAACL2006.

Zha H. 2002. *Generic Summarization and Key Phrase Extraction using Mutual Reinforcement Principle and Sentence Clustering*. In Proceedings of SIGIR2002.

End-to-End Coreference Resolution via Hypergraph Partitioning

Jie Cai and **Michael Strube**
Natural Language Processing Group
Heidelberg Institute for Theoretical Studies gGmbH
(jie.cai|michael.strube)@h-its.org

Abstract

We describe a novel approach to coreference resolution which implements a global decision via hypergraph partitioning. In constrast to almost all previous approaches, we do not rely on separate classification and clustering steps, but perform coreference resolution globally in one step. Our hypergraph-based global model implemented within an end-to-end coreference resolution system outperforms two strong baselines (Soon et al., 2001; Bengtson & Roth, 2008) using system mentions only.

1 Introduction

Coreference resolution is the task of grouping mentions of entities into sets so that all mentions in one set refer to the same entity. Most recent approaches to coreference resolution divide this task into two steps: (1) a classification step which determines whether a pair of mentions is coreferent or which outputs a confidence value, and (2) a clustering step which groups mentions into entities based on the output of step 1.

The classification steps of most approaches vary in the choice of the classifier (e.g. decision tree classifiers (Soon et al., 2001), maximum entropy classification (Luo et al., 2004), SVM classifiers (Rahman & Ng, 2009)) and the number of features used (Soon et al. (2001) employ a set of twelve simple but effective features while e.g., Ng & Cardie (2002) and Bengtson & Roth (2008) devise much richer feature sets).

The clustering step exhibits much more variation: Local variants utilize a closest-first decision (Soon et al., 2001), where a mention is resolved to its closest possible antecedent, or a best-first decision (Ng & Cardie, 2002), where a mention is resolved to its most confident antecedent (based on the confidence value returned by step 1). Global variants attempt to consider all possible clustering possibilites by creating and searching a *Bell tree* (Luo et al., 2004), by learning the optimal search strategy itself (Daumé III & Marcu, 2005), by building a graph representation and applying graph clustering techniques (Nicolae & Nicolae, 2006), or by employing integer linear programming (Klenner, 2007; Denis & Baldridge, 2009). Since these methods base their global clustering step on a local pairwise model, some global information which could have guided step 2 is already lost. The twin-candidate model (Yang et al., 2008) replaces the pairwise model by learning preferences between two antecedent candidates in step 1 and applies tournament schemes instead of the clustering in step 2.

There is little work which deviates from this two-step scheme. Culotta et al. (2007) introduce a first-order probabilistic model which implements features over sets of mentions and thus operates directly on entities.

In this paper we describe a novel approach to coreference resolution which avoids the division into two steps and instead performs a global decision in one step. We represent a document as a hypergraph, where the vertices denote mentions and the edges denote relational features between mentions. Coreference resolution is performed globally in one step by partitioning the hypergraph into subhypergraphs so that all mentions in one subhypergraph refer to the same entity. Our model out-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 143–151,
Beijing, August 2010

performs two strong baselines, Soon et al. (2001) and Bengtson & Roth (2008).

Soon et al. (2001) developed an end-to-end coreference resolution system for the MUC data, i.e., a system which processes raw documents as input and produces annotated ones as output. However, with the advent of the ACE data, many systems either evaluated only true mentions, i.e. mentions which are included in the annotation, the so-called key, or even received true information for mention boundaries, heads of mentions and mention type (Culotta et al., 2007, inter alia). While these papers report impressive results it has been concluded that this experimental setup simplifies the task and leads to an unrealistic surrogate for the coreference resolution problem (Stoyanov et al., 2009, p.657, p660). We argue that the field should move towards a realistic setting using system mentions, i.e. automatically determined mention boundaries and types. In this paper we report results using our end-to-end coreference resolution system, COPA, without relying on unrealistic assumptions.

2 Related Work

Soon et al. (2001) transform the coreference resolution problem straightforwardly into a pairwise classification task making it accessible to standard machine learning classifiers. They use a set of twelve powerful features. Their system is based solely on information of the mention pair anaphor and antecedent. It does not take any information of other mentions into account. However, it turned out that it is difficult to improve upon their results just by applying a more sophisticated learning method and without improving the features. We use a reimplementation of their system as first baseline. Bengtson & Roth (2008) push this approach to the limit by devising a much more informative feature set. They report the best results to date on the ACE 2004 data using true mentions. We use their system combined with our preprocessing components as second baseline.

Luo et al. (2004) perform the clustering step within a Bell tree representation. Hence their system theoretically has access to all possible outcomes making it a potentially global system. However, the classification step is still based on a pairwise model. Also since the search space in the Bell tree is too large they have to apply search heuristics. Hence, their approach loses much of the power of a truly global approach.

Culotta et al. (2007) introduce a first-order probabilistic model which implements features over sets of mentions. They use four features for their first-order model. The first is an enumeration over *pairs* of noun phrases. The second is the output of a *pairwise* model. The third is the cluster size. The fourth counts mention type, number and gender in each cluster. Still, their model is based mostly on information about pairs of mentions. They assume true mentions as input. It is not clear whether the improvement in results translates to system mentions.

Nicolae & Nicolae (2006) describe a graph-based approach which superficially resembles our approach. However, they still implement a two step coreference resolution approach and apply the global graph-based model only to step 2. They report considerable improvements over state-of-the-art systems including Luo et al. (2004). However, since they not only change the clustering strategy but also the features for step 1, it is not clear whether the improvements are due to the graph-based clustering technique. We, instead, describe a graph-based approach which performs classification and clustering in one step. We compare our approach with two competitive systems using the same feature sets.

3 COPA: Coreference Partitioner

The COPA system consists of learning modules which learn hyperedge weights from the training data, and resolution modules which create a hypergraph representation for the testing data and perform partitioning to produce subhypergraphs, each of which represents an entity. An example analysis of a short document involving the two entities, BARACK OBAMA and NICOLAS SARKOZY illustrates how COPA works.

[US President Barack Obama] came to Toronto today.
[Obama] discussed the financial crisis with [President Sarkozy].
[He] talked to him [him] about the recent downturn of the European markets.
[Barack Obama] will leave Toronto tomorrow.

A hypergraph (Figure (1a)) is built for this document based on three features. Two hyperedges denote the feature *partial string match*, {*US President Barack Obama, Barack Obama, Obama*} and {*US President Barack Obama, President Sarkozy*}. One hyperedge denotes the feature *pronoun match*, {*he, him*}. Two hyperedges denote the feature *all speak*, {*Obama, he*} and {*President Sarkozy, him*}.

On this initial representation, a spectral clustering technique is applied to find two partitions which have the strongest within-cluster connections and the weakest between-clusters relations. The cut found is called *Normalized Cut*, which avoids trivial partitions frequently output by the min-cut algorithm. The two output subhypergraphs (Figure (1b)) correspond to two resolved entities shown on both sides of the bold dashed line. In real cases, recursive cutting is applied to all the subhypergraphs resulting from previous steps, until a stopping criterion is reached.

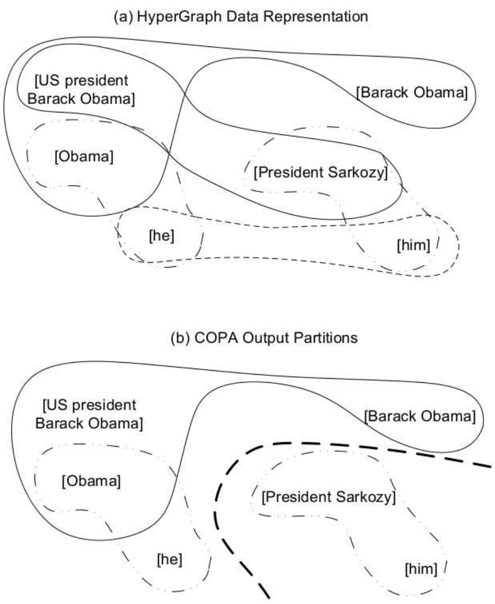


Figure 1: Hypergraph-based representation

3.1 HyperEdgeLearner

COPA needs training data only for computing the hyperedge weights. Hyperedges represent features. Each hyperedge corresponds to a feature instance modeling a simple relation between two or more mentions. This leads to initially overlapping sets of mentions. Hyperedges are assigned weights which are calculated based on the training data as the percentage of the initial edges (as illustrated in Figure (1a)) being in fact coreferent. The weights for some of Soon et al. (2001)'s features learned from the ACE 2004 training data are given in Table 1.

Edge Name	Weight
Alias	0.777
StrMatch_Pron	0.702
Appositive	0.568
StrMatch_Npron	0.657
ContinuousDistAgree	0.403

Table 1: Hyperedge weights for ACE 2004 data

3.2 Coreference Resolution Modules

Unlike pairwise models, COPA processes a document globally in one step, taking care of the preference information among all the mentions at the same time and clustering them into sets directly. A raw document is represented as a single hypergraph with multiple edges. The hypergraph resolver partitions the simple hypergraph into several subhypergraphs, each corresponding to one set of coreferent mentions (see e.g. Figure (1b) which contains two subhypergraphs).

3.2.1 HGModelBuilder

A single document is represented in a hypergraph with basic relational features. Each hyperedge in a graph corresponds to an instance of one of those features with the weight assigned by the *HyperEdgeLearner*. Instead of connecting nodes with the target relation as usually done in graph models, COPA builds the graph directly out of a set of low dimensional features without any assumptions for a distance metric.

3.2.2 HGResolver

In order to partition the hypergraph we adopt a spectral clustering algorithm. Spectral clustering techniques use information obtained from the eigenvalues and eigenvectors of the graph Laplacian to cluster the vertices. They are simple to implement and reasonably fast and have been shown to frequently outperform traditional clustering algorithms such as k-means. These techniques have

Algorithm 1 R2 partitioner

Note: { $L = I - D_v{}^{-\frac{1}{2}} HWD_e{}^{-1} H^T D_v{}^{-\frac{1}{2}}$ }
Note: { $Ncut(S) := vol\partial S(\frac{1}{volS} + \frac{1}{volS^c})$}
input: target hypergraph HG, predefined $\alpha^\star$
Given a HG, construct its D_v, H, W and D_e
Compute L for HG
Solve the L for the second smallest eigenvector V_2
for each splitting point in V_2 **do**
  calculate $Ncut_i$
end for
Choose the splitting point with $\min_i(Ncut_i)$
Generate two subHGs
if $\min_i(Ncut_i) < \alpha^*$ **then**
  for each subHG **do**
    Bi-partition the subHG with the *R2 partitioner*
  end for
else
  Output the current subHG
end if
output: partitioned HG

many applications, e.g. image segmentation (Shi & Malik, 2000).

We adopt two variants of spectral clustering, *recursive 2-way partitioning (R2 partitioner)* and *flat-K partitioning*. Since flat-K partitioning did not perform as well we focus here on recursive 2-way partitioning. In contrast to flat-K partitioning, this method does not need any information about the number of target sets. Instead a stopping criterion $\alpha^\star$ has to be provided. $\alpha^\star$ is adjusted on development data (see Algorithm 1).

In order to apply spectral clustering to hypergraphs we follow Agarwal et al. (2005). All experimental results are obtained using symmetric Laplacians (L_{sym}) (von Luxburg, 2007).

Given a hypergraph *HG*, a set of matrices is generated. D_v and D_e denote the diagonal matrices containing the vertex and hyperedge degrees respectively. $|V| \times |E|$ matrix H represents the *HG* with the entries $h(v, e) = 1$ if $v \in e$ and 0 otherwise. H^T is the transpose of H. W is the diagonal matrix with the edge weights. S is one of the subhypergraphs generated from a cut in the *HG*, where $Ncut(S)$ is the cut's value.

Using Normalized Cut does not generate singleton clusters, hence a heuristic singleton detection strategy is used in COPA. We apply a threshold β to each node in the graph. If a node's degree is below the threshold, the node will be removed.

3.3 Complexity of HGResolver

Since edge weights are assigned using simple descriptive statistics, the time HGResolver needs for building the graph Laplacian matrix is insubstantial. For eigensolving, we use an open source library provided by the Colt project[1]which implements a Householder-QL algorithm to solve the eigenvalue decomposition. When applied to the symmetric graph Laplacian, the complexity of the eigensolving is given by $O(n^3)$, where n is the number of mentions in a hypergraph. Since there are only a few hundred mentions per document in our data, this complexity is not an issue (spectral clustering gets problematic when applied to millions of data points).

4 Features

The *HGModelBuilder* allows hyperedges with a degree higher than two to grow throughout the building process. This type of edge is *mergeable*. Edges with a degree of two describe pairwise relations. Thus these edges are *non-mergeable*. This way any kind of relational features can be incorporated into the hypergraph model.

Features are represented as types of hyperedges (in Figure (1b) the two hyperedges marked by "–··" are of the same type). Any realized edge is an instance of the corresponding edge type. All instances derived from the same type have the same weight, but they may get reweighted by the distance feature (Section 4.4).

In the following Subsections we describe the features used in our experiments. We use the entire set for obtaining the final results. We restrict ourselves to Soon et al. (2001)'s features when we compare our system with theirs in order to assess the impact of our model regardless of features (we use features 1., 2., 3., 6., 7., 11., 13.).

4.1 Hyperedges With a Degree > 2

High degree edges are the particular property of the hypergraph which allows to include all types of relational features into our model. The edges are built through pairwise relations and, if consistent, get incrementally merged into larger edges.

[1] http://acs.lbl.gov/~hoschek/colt/

High degree edges are not sensitive to positional information from the documents.

(1) StrMatch_Npron & (2) StrMatch_Pron: After discarding stop words, if the strings of mentions completely match and are not pronouns, they are put into edges of the *StrMatch_Npron* type. When the matched mentions are pronouns, they are put into the *StrMatch_Pron* type edges.

(3) Alias: After discarding stop words, if mentions are aliases of each other (i.e. proper names with partial match, full names and acronyms of organizations, etc.), they are put into edges of the *Alias* type.

(4) Synonym: If, according to WordNet, mentions are synonymous, they are put into an edge of the *Synonym* type.

(5) AllSpeak: Mentions which appear within a window of two words of a verb meaning *to say* form an edge of the *AllSpeak* type.

(6) Agreement: If mentions agree in *Gender*, *Number* and *Semantic Class* they are put in edges of the *Agreement* type. Because *Gender*, *Number* and *Semantic Class* are strong negative coreference indicators – in contrast to e.g. *StrMatch* – and hence weak positive features, they are combined into the one feature *Agreement*.

4.2 Hyperedges With a Degree = 2

Features which have been used by pairwise models are easily integrated into the hypergraph model by generating edges with only two vertices. Information sensitive to relative distance is represented by pairwise edges.

(7) Apposition & (8) RelativePronoun: If two mentions are in a appositive structure, they are put in an edge of type *Apposition*. If the latter mention is a relative pronoun, the mentions are put in an edge of type *RelativePronoun*.

(9) HeadModMatch: If the syntactic heads of two mentions match, and if their modifiers do not contradict each other, the mentions are put in an edge of type *HeadModMatch*.

(10) SubString: If a mention is the substring of another one, they are put into an edge of type *SubString*.

4.3 MentionType and EntityType

In our model **(11) mention type** can only reasonably be used when it is conjoined with other features, since mention type itself describes an attribute of single mentions. In COPA, it is conjoined with other features to form hyperedges, e.g. the *StrMatch_Pron* edge. We use the same strategy to represent **(12) entity type**.

4.4 Distance Weights

Our hypergraph model does not have any obvious means to encode distance information. However, the distance between two mentions plays an important role in coreference resolution, especially for resolving pronouns. We do not encode distance as feature, because this would introduce many two-degree-hyperedges which would be computationally very expensive without much gain in performance. Instead, we use distance to reweight two-degree-hyperedges, which are sensitive to positional information.

We experimented with two types of distance weights: One is **(13) sentence distance** as used in Soon et al. (2001)'s feature set, while the other is **(14) compatible mentions distance** as introduced by Bengtson & Roth (2008).

5 Experiments

We compare COPA's performance with two implementations of pairwise models. The first baseline is the BART (Versley et al., 2008) reimplementation of Soon et al. (2001), with few but effective features. Our second baseline is Bengtson & Roth (2008), which exploits a much larger feature set while keeping the machine learning approach simple. Bengtson & Roth (2008) show that their system outperforms much more sophisticated machine learning approaches such as Culotta et al. (2007), who reported the best results on true mentions before Bengtson & Roth (2008). Hence, Bengtson & Roth (2008) seems to be a reasonable competitor for evaluating COPA.

In order to report realistic results, we neither assume true mentions as input nor do we evaluate only on true mentions. Instead, we use an inhouse mention tagger for automatically extracting mentions.

5.1 Data

We use the MUC6 data (Chinchor & Sundheim, 2003) with standard training/testing divisions (30/30) as well as the MUC7 data (Chinchor, 2001) (30/20). Since we do not have access to the official ACE testing data (only available to ACE participants), we follow Bengtson & Roth (2008) for dividing the ACE 2004 English training data (Mitchell et al., 2004) into training, development and testing partitions (268/76/107). We randomly split the 252 ACE 2003 training documents (Mitchell et al., 2003) using the same proportions into training, development and testing (151/38/63). The systems were tuned on development and run only once on testing data.

5.2 Mention Tagger

We implement a classification-based mention tagger, which tags each NP chunk as ACE mention or not, with neccessary post-processing for embedded mentions. For the ACE 2004 testing data, we cover 75.8% of the heads with 73.5% accuracy.

5.3 Evaluation Metrics

We evaluate COPA with three coreference resolution evaluation metrics: the B^3-algorithm (Bagga & Baldwin, 1998), the *CEAF*-algorithm (Luo, 2005), and, for the sake of completeness, the *MUC*-score (Vilain et al., 1995).

Since the *MUC*-score does not evaluate singleton entities, it only partially evaluates the performance for ACE data, which includes singleton entities in the keys. The B^3-algorithm (Bagga & Baldwin, 1998) addresses this problem of the *MUC*-score by conducting calculations based on mentions instead of coreference relations. However, another problematic issue emerges when system mentions have to be dealt with: B^3 assumes the mentions in the key and in the response to be identical, which is unlikely when a mention tagger is used to create system mentions. The *CEAF*-algorithm aligns entities in key and response by means of a similarity metric, which is motivated by B^3's shortcoming of using one entity multiple times (Luo, 2005). However, although *CEAF* theoretically does not require to have the same number of mentions in key and response, the algorithm still cannot be directly applied to end-to-end coreference resolution systems, because the similarity metric is influenced by the number of mentions in key and response.

Hence, both the B^3- and *CEAF*-algorithms have to be extended to deal with system mentions which are not in the key and true mentions not extracted by the system, so called *twinless mentions* (Stoyanov et al., 2009). Two variants of the B^3-algorithm are proposed by Stoyanov et al. (2009), B^3_{all} and B^3_0. B^3_{all} tries to assign intuitive precision and recall to the twinless system mentions and twinless key mentions, while keeping the size of the system mention set and the key mention set unchanged (which are different from each other). For twinless mentions, B^3_{all} discards twinless key mentions for precision and twinless system mentions for recall. Discarding parts of the key mentions, however, makes the fair comparison of precision values difficult. B^3_0 produces counter-intuitive precision by discarding all twinless system mentions. Although it penalizes the recall of all twinless key mentions, so that the F-scores are balanced, it is still too lenient (for further analyses see Cai & Strube (2010)).

We devise two variants of the B^3- and *CEAF*-algorithms, namely B^3_{sys} and *CEAF*$_{sys}$. For computing precision, the algorithms put all twinless true mentions into the response even if they were not extracted. All twinless system mentions which were deemed not coreferent are discarded. Only twinless system mentions which were mistakenly resolved are put into the key. Hence, the system is penalized for resolving mentions not found in the key. For recall the algorithms only consider mentions from the original key by discarding all the twinless system mentions and putting twinless true mentions into the response as singletons (algorithm details, simulations and comparison of different systems and metrics are provided in Cai & Strube (2010)). For *CEAF*$_{sys}$, ϕ_3 (Luo, 2005) is used. B^3_{sys} and *CEAF*$_{sys}$ report results for end-to-end coreference resolution systems adequately.

5.4 Baselines

We compare COPA's performance with two baselines: *SOON* – the BART (Versley et al., 2008) reimplementation of Soon et al. (2001) – and

		SOON			COPA with R2 partitioner				
		R	P	F	R	P	F	$\alpha^\star$	β
MUC	MUC6	59.4	67.9	63.4	62.8	66.4	**64.5**	0.08	0.03
	MUC7	52.3	67.1	58.8	55.2	66.1	**60.1**	0.05	0.01
	ACE 2003	56.7	75.8	64.9	60.8	75.1	**67.2**	0.07	0.03
	ACE 2004	50.4	67.4	57.7	54.1	67.3	**60.0**	0.05	0.04
B^3_{sys}	MUC6	53.1	78.9	63.5	56.4	76.3	**64.1**	0.08	0.03
	MUC7	49.8	80.0	61.4	53.3	76.1	**62.7**	0.05	0.01
	ACE 2003	66.9	87.7	75.9	71.5	83.3	**77.0**	0.07	0.03
	ACE 2004	64.7	85.7	73.8	67.3	83.4	**74.5**	0.07	0.03
CEAF$_{sys}$	MUC6	56.9	53.0	54.9	62.2	57.5	**59.8**	0.08	0.03
	MUC7	57.3	54.3	55.7	58.3	54.2	56.2	0.06	0.01
	ACE 2003	71.0	68.7	69.8	71.1	68.3	69.7	0.07	0.03
	ACE 2004	67.9	65.2	66.5	68.5	65.5	67.0	0.07	0.03

Table 3: *SOON* vs. COPA R2 (*SOON* features, system mentions, bold indicates significant improvement in F-score over *SOON* according to a paired-t test with $p < 0.05$)

	SOON			*B&R*		
	R	P	F	R	P	F
B^3_{sys}	64.7	85.7	73.8	66.3	85.8	74.8

Table 2: Baselines on ACE 2004

B&R – Bengtson & Roth (2008)[2]. All systems share BART's preprocessing components and our in-house ACE mention tagger.

In Table 2 we report the performance of *SOON* and *B&R* on the ACE 2004 testing data using the BART preprocessing components and our in-house ACE mention tagger. For evaluation we use B^3_{sys} only, since Bengtson & Roth (2008)'s system does not allow to easily integrate *CEAF*.

B&R considerably outperforms *SOON* (we cannot compute statistical significance, because we do not have access to results for single documents in *B&R*). The difference, however, is not as big as we expected. Bengtson & Roth (2008) reported very good results when using true mentions. For evaluating on system mentions, however, they were using a too lenient variant of B^3 (Stoyanov et al., 2009) which discards all twinless mentions. When replacing this with B^3_{sys} the difference between *SOON* and *B&R* shrinks.

5.5 Results

In both comparisons, COPA uses the same features as the corresponding baseline system.

5.5.1 COPA vs. *SOON*

In Table 3 we compare the *SOON*-baseline with COPA using the R2 partitioner (parameters $\alpha^\star$ and β optimized on development data). Even though COPA and *SOON* use the same features, COPA consistently outperforms *SOON* on all data sets using all evaluation metrics. With the exception of the MUC7, the ACE 2003 and the ACE 2004 data evaluated with *CEAF*$_{sys}$, all of COPA's improvements are statistically significant. When evaluated using *MUC* and B^3_{sys}, COPA with the R2 partitioner boosts recall in all datasets while losing in precision. This shows that global hypergraph partitioning models the coreference resolution task more adequately than Soon et al. (2001)'s local model – even when using the very same features.

5.5.2 COPA vs. *B&R*

In Table 4 we compare the *B&R* system (using our preprocessing components and mention tagger), and COPA with the R2 partitioner using *B&R* features. COPA does not use the learned features from *B&R*, as this would have implied to embed a pairwise coreference resolution system in COPA. We report results for ACE 2003 and ACE 2004. The parameters are optimized on the ACE 2004 data. COPA with the R2 partitioner outperforms *B&R* on both datasets (we cannot compute statistical significance, because we do not have access to results for single documents in *B&R*). Bengtson & Roth (2008) developed their system on ACE 2004 data and never exposed it to ACE 2003 data. We suspect that the relatively poor result of *B&R* on ACE 2003 data is caused by overfitting to ACE

[2] `http://l2r.cs.uiuc.edu/~cogcomp/asoftware.php?skey=FLBJCOREF`

		B&R			COPA with R2 partitioner		
		R	P	F	R	P	F
B^3_{sys}	ACE 2003	56.4	97.3	71.4	70.3	86.5	77.5
	ACE 2004	66.3	85.8	74.8	68.4	84.4	75.6

Table 4: *B&R* vs. COPA R2 (*B&R* features, system mentions)

2004. Again, COPA gains in recall and loses in precision. This shows that COPA is a highly competetive system as it outperforms Bengtson & Roth (2008)'s system which has been claimed to have the best performance on the ACE 2004 data.

5.5.3 Running Time

On a machine with 2 AMD Opteron CPUs and 8 GB RAM, COPA finishes preprocessing, training and partitioning the ACE 2004 dataset in 15 minutes, which is slightly faster than our duplicated *SOON* baseline.

6 Discussion and Outlook

Most previous attempts to solve the coreference resolution task globally have been hampered by employing a local pairwise model in the classification step (step 1) while only the clustering step realizes a global approach, e.g. Luo et al. (2004), Nicolae & Nicolae (2006), Klenner (2007), Denis & Baldridge (2009), lesser so Culotta et al. (2007). It has been also observed that improvements in performance on true mentions do not necessarily translate into performance improvements on system mentions (Ng, 2008).

In this paper we describe a coreference resolution system, COPA, which implements a global decision in one step via hypergraph partitioning. COPA looks at the whole graph at once which enables it to outperform two strong baselines (Soon et al., 2001; Bengtson & Roth, 2008). COPA's hypergraph-based strategy can be taken as a general preference model, where the preference for one mention depends on information on all other mentions.

We follow Stoyanov et al. (2009) and argue that evaluating the performance of coreference resolution systems on true mentions is unrealistic. Hence we integrate an ACE mention tagger into our system, tune the system towards the real task, and evaluate only using system mentions. While Ng (2008) could not show that superior models achieved superior results on system mentions, COPA was able to outperform Bengtson & Roth (2008)'s system which has been claimed to achieve the best performance on the ACE 2004 data (using true mentions, Bengtson & Roth (2008) did not report any comparison with other systems using system mentions).

An error analysis revealed that there were some cluster-level inconsistencies in the COPA output. Enforcing this consistency would require a global strategy to propagate constraints, so that constraints can be included in the hypergraph partitioning properly. We are currently exploring constrained clustering, a field which has been very active recently (Basu et al., 2009). Using constrained clustering methods may allow us to integrate negative information as constraints instead of combining several weak positive features to one which is still weak (e.g. our *Agreement* feature). For an application of constrained clustering to the related task of database record linkage, see Bhattacharya & Getoor (2009).

Graph models cannot deal well with positional information, such as distance between mentions or the sequential ordering of mentions in a document. We implemented distance as weights on hyperedges which resulted in decent performance. However, this is limited to pairwise relations and thus does not exploit the power of the high degree relations available in COPA. We expect further improvements, once we manage to include positional information directly.

Acknowledgements. This work has been funded by the Klaus Tschira Foundation, Heidelberg, Germany. The first author has been supported by a HITS PhD. scholarship. We would like to thank Byoung-Tak Zhang for bringing hypergraphs to our attention and Èva Mújdricza-Maydt for implementing the mention tagger. Finally we would like to thank our colleagues in the HITS NLP group for providing us with useful comments.

References

Agarwal, Sameer, Jonwoo Lim, Lihi Zelnik-Manor, Pietro Perona, David Kriegman & Serge Belongie (2005). Beyond pairwise clustering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2, pp. 838–845.

Bagga, Amit & Breck Baldwin (1998). Algorithms for scoring coreference chains. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, Granada, Spain, 28–30 May 1998, pp. 563–566.

Basu, Sugato, Ian Davidson & Kiri L. Wagstaff (Eds.) (2009). *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Boca Raton, Flo.: CRC Press.

Bengtson, Eric & Dan Roth (2008). Understanding the value of features for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Waikiki, Honolulu, Hawaii, 25-27 October 2008, pp. 294–303.

Bhattacharya, Indrajit & Lise Getoor (2009). Collective relational clustering. In S. Basu, I. Davidson & K. Wagstaff (Eds.), *Constrained Clustering: Advances in Algorithms, Theory, and Applications*, pp. 221–244. Boca Raton, Flo.: CRC Press.

Cai, Jie & Michael Strube (2010). Evaluation metrics for end-to-end coreference resolution systems. In *Proceedings of the SIGdial 2010 Conference: The 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Tokyo, Japan, 24–25 September 2010. To appear.

Chinchor, Nancy (2001). *Message Understanding Conference (MUC) 7*. LDC2001T02, Philadelphia, Penn: Linguistic Data Consortium.

Chinchor, Nancy & Beth Sundheim (2003). *Message Understanding Conference (MUC) 6*. LDC2003T13, Philadelphia, Penn: Linguistic Data Consortium.

Culotta, Aron, Michael Wick & Andrew McCallum (2007). First-order probabilistic models for coreference resolution. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, N.Y., 22–27 April 2007, pp. 81–88.

Daumé III, Hal & Daniel Marcu (2005). A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proceedings of the Human Language Technology Conference and the 2005 Conference on Empirical Methods in Natural Language Processing*, Vancouver, B.C., Canada, 6–8 October 2005, pp. 97–104.

Denis, Pascal & Jason Baldridge (2009). Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural*, 42:87–96.

Klenner, Manfred (2007). Enforcing consistency on coreference sets. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*, Borovets, Bulgaria, 27–29 September 2007, pp. 323–328.

Luo, Xiaoqiang (2005). On coreference resolution performance metrics. In *Proceedings of the Human Language Technology Conference and the 2005 Conference on Empirical Methods in Natural Language Processing*, Vancouver, B.C., Canada, 6–8 October 2005, pp. 25–32.

Luo, Xiaoqiang, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla & Salim Roukos (2004). A mention-synchronous coreference resolution algorithm based on the Bell Tree. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, Barcelona, Spain, 21–26 July 2004, pp. 136–143.

Mitchell, Alexis, Stephanie Strassel, Shudong Huang & Ramez Zakhary (2004). *ACE 2004 Multilingual Training Corpus*. LDC2005T09, Philadelphia, Penn.: Linguistic Data Consortium.

Mitchell, Alexis, Stephanie Strassel, Mark Przybocki, JK Davis, George Doddington, Ralph Grishman, Adam Meyers, Ada Brunstain, Lisa Ferro & Beth Sundheim (2003). *TIDES Extraction (ACE) 2003 Multilingual Training Data*. LDC2004T09, Philadelphia, Penn.: Linguistic Data Consortium.

Ng, Vincent (2008). Unsupervised models for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Waikiki, Honolulu, Hawaii, 25-27 October 2008, pp. 640–649.

Ng, Vincent & Claire Cardie (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, Penn., 7–12 July 2002, pp. 104–111.

Nicolae, Cristina & Gabriel Nicolae (2006). BestCut: A graph algorithm for coreference resolution. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia, 22–23 July 2006, pp. 275–283.

Rahman, Altaf & Vincent Ng (2009). Supervised models for coreference resolution. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, 6-7 August 2009, pp. 968–977.

Shi, Jianbo & Jitendra Malik (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Soon, Wee Meng, Hwee Tou Ng & Daniel Chung Yong Lim (2001). A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Stoyanov, Veselin, Nathan Gilbert, Claire Cardie & Ellen Riloff (2009). Conundrums in noun phrase coreference resolution: Making sense of the state-of-the-art. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing*, Singapore, 2–7 August 2009, pp. 656–664.

Versley, Yannick, Simone Paolo Ponzetto, Massimo Poesio, Vladimir Eidelman, Alan Jern, Jason Smith, Xiaofeng Yang & Alessandro Moschitti (2008). BART: A modular toolkit for coreference resolution. In *Companion Volume to the Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, Columbus, Ohio, 15–20 June 2008, pp. 9–12.

Vilain, Marc, John Burger, John Aberdeen, Dennis Connolly & Lynette Hirschman (1995). A model-theoretic coreference scoring scheme. In *Proceedings of the 6th Message Understanding Conference (MUC-6)*, pp. 45–52. San Mateo, Cal.: Morgan Kaufmann.

von Luxburg, Ulrike (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416.

Yang, Xiaofeng, Jian Su & Chew Lim Tan (2008). A twin-candidate model for learning-based anaphora resolution. *Computational Linguistics*, 34(3):327–356.

Exploiting Background Knowledge for Relation Extraction

Yee Seng Chan and **Dan Roth**
University of Illinois at Urbana-Champaign
{chanys,danr}@illinois.edu

Abstract

Relation extraction is the task of recognizing semantic relations among entities. Given a particular sentence supervised approaches to Relation Extraction employed feature or kernel functions which usually have a single sentence in their scope. The overall aim of this paper is to propose methods for using knowledge and resources that are external to the target sentence, as a way to improve relation extraction. We demonstrate this by exploiting background knowledge such as relationships among the target relations, as well as by considering how target relations relate to some existing knowledge resources. Our methods are general and we suggest that some of them could be applied to other NLP tasks.

1 Introduction

Relation extraction (RE) is the task of detecting and characterizing semantic relations expressed between entities in text. For instance, given the sentence "Cone, a Kansas City native, was originally signed by the Royals and broke into the majors with the team.", one of the relations we might want to extract is the *employment* relation between the pair of entity mentions "Cone" and "Royals". RE is important for many NLP applications such as building an ontology of entities, biomedical information extraction, and question answering.

Prior work have employed diverse approaches towards resolving the task. One approach is to build supervised RE systems using sentences annotated with entity mentions and predefined target relations. When given a new sentence, the RE system has to detect and disambiguate the presence of any predefined relations that might exist between each of the mention pairs in the sentence. In building these systems, researchers used a wide variety of features (Kambhatla, 2004; Zhou et al., 2005; Jiang and Zhai, 2007). Some of the common features used to analyze the target sentence include the words appearing in the sentence, their part-of-speech (POS) tags, the syntactic parse of the sentence, and the dependency path between the pair of mentions. In a related line of work, researchers have also proposed various kernel functions based on different structured representations (e.g. dependency or syntactic tree parses) of the target sentences (Bunescu and Mooney, 2005; Zhou et al., 2007; Zelenko et al., 2003; Zhang et al., 2006). Additionally, researchers have tried to automatically extract examples for supervised learning from resources such as Wikipedia (Weld et al., 2008) and databases (Mintz et al., 2009), or attempted *open* information extraction (IE) (Banko et al., 2007) to extract all possible relations.

In this work, we focus on supervised RE. In prior work, the feature and kernel functions employed are usually restricted to being defined on the various representations (e.g. lexical or structural) of the target sentences. However, in recognizing relations, humans are not thus constrained and rely on an abundance of implicit *world knowledge* or *background information*. What quantifies as world or background knowledge is rarely explored in the RE literature and we do not attempt to provide complete nor precise definitions in this paper. However, we show that by considering the relationship between our relations of interest, as

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 152–160,
Beijing, August 2010

well as how they relate to some existing knowledge resources, we improve the performance of RE. Specifically, the contributions of this paper are the following:

- When our relations of interest are clustered or organized in a hierarchical ontology, we show how to use this information to improve performance. By defining appropriate constraints between the predictions of relations at different levels of the hierarchy, we obtain globally coherent predictions as well as improved performance.

- Coreference is a generic relationship that might exists among entity mentions and we show how to exploit this information by assuming that co-referring mentions have no other interesting relations. We capture this intuition by using coreference information to constraint the predictions of a RE system.

- When characterizing the relationship between a pair of mentions, one can use a large encyclopedia such as Wikipedia to infer more knowledge about the two mentions. In this work, after probabilistically mapping mentions to their respective Wikipedia pages, we check whether the mentions are related. Another generic relationship that might exists between a pair of mentions is whether they have a *parent-child* relation and we use this as additional information.

- The sparsity of features (especially lexical features) is a common problem for supervised systems. In this work, we show that one can make fruitful use of unlabeled data, by using word clusters automatically gathered from unlabeled texts as a way of generalizing the lexical features.

- We combine the various relational predictions and background knowledge through a global inference procedure, which we formalize via an Integer Linear Programming (ILP) framework as a constraint optimization problem (Roth and Yih, 2007). This allows us to easily incorporate various constraints that encode the background knowledge.

Roth and Yih (2004) develop a relation extraction approach that exploits constraints among entity types and the relations allowed among them. We extend this view significantly, within a similar computational framework, to exploit relations among target relations, background information and world knowledge, as a way to improve relation extraction and make globally coherent predictions.

In the rest of this paper, we first describe the features used in our basic RE system in Section 2. We then describe how we make use of background knowledge in Section 3. In Section 4, we show our experimental results and perform analysis in Section 5. In Section 6, we discuss related work, before concluding in Section 7.

2 Relation Extraction System

In this section, we describe the features used in our basic relation extraction (RE) system. Given a pair of mentions m_1 and m_2 occurring within the same sentence, the system predicts whether any of the predefined relation holds between the two mentions. Since relations are usually asymmetric in nature, hence in all of our experiments, unless otherwise stated, we distinguish between the argument ordering of the two mentions. For instance, we consider m_1*:emp-org:*m_2 and m_2*:emp-org:*m_1 to be distinct relation types.

Most of the features used in our system are based on the work in (Zhou et al., 2005). In this paper, we propose some new collocation features inspired by word sense disambiguation (WSD). We give an overview of the features in Table 1. Due to space limitations, we only describe the collocation features and refer the reader to (Zhou et al., 2005) for the rest of the features.

2.1 Collocation Features

Following (Zhou et al., 2005), we use a single word to represent the head word of a mention. Since single words might be ambiguous or polysemous, we incorporate local collocation features which were found to be very useful for WSD. Given the head word hw_m of a mention *m*, the collocation feature $C_{i,j}$ refers to the sequence of tokens in the immediate context of hw_m. The offsets *i* and *j* denote the position (relative to hw_m)

Category	Feature
Lexical	hw of m_1
	hw of m_2
	hw of m_1, m_2
	BOW in m_1
	BOW in m_2
	single word between m_1, m_2
	BOW in between m_1, m_2
	bigrams in between m_1, m_2
	first word in between m_1, m_2
	last word in between m_1, m_2
Collocations	$C_{-1,-1}$, $C_{+1,+1}$
	$C_{-2,-1}$, $C_{-1,+1}$, $C_{+1,+2}$
Structural	m_1-in-m_2
	m_2-in-m_1
	#mentions between m_1, m_2
	any word between m_1, m_2
M-lvl	M-lvl of m_1, m_2
and	m_1, m_2 E-maintype
E-type	m_1, m_2 E-subtype
	m_1, m_2 M-lvl and E-maintype
	m_1, m_2 M-lvl and E-subtype
	m_1, m_2 E-subtype and m_1-in-m_2
	m_1, m_2 E-subtype and m_2-in-m_1
Dependency	path between m_1, m_2
	bag-of dep labels between m_1, m_2
	hw of m_1 and dep-parent
	hw of m_2 and dep-parent

Table 1: Features in the basic RE system. The abbreviations are as follows. hw: head word, M-lvl: mention level, E-type: entity type, dep-parent: the word's parent in the dependency tree.

of the first and last token of the sequence respectively. For instance, $C_{-1,+1}$ denotes a sequence of three tokens, consisting of the single token on the immediate left of hw_m, the token hw_m itself, and the single token on the immediate right of hw_m. For each mention, we extract 5 features: $C_{-1,-1}$, $C_{+1,+1}$, $C_{-2,-1}$, $C_{-1,+1}$, and $C_{+1,+2}$.

3 Using Background Knowledge

Now we describe how we inject additional knowledge into our relation extraction system.

3.1 Hierarchy of Relations

When our relations of interest are arranged in a hierarchical structure, one should leverage this information to learn more accurate relation predictors. For instance, assume that our relations are arranged in a two-level hierarchy and we learn two classifiers, one for disambiguating between the first level *coarse-grained* relations, and another for disambiguating between the second level *fine-grained* relations.

Since there are a lot more fine-grained relation types than coarse-grained relation types, we propose using the coarse-grained predictions which should intuitively be more reliable, to improve the fine-grained predictions. We show how to achieve this through defining appropriate constraints between the coarse-grained and fine-grained relations, which can be enforced through the Constrained Conditional Models framework (aka ILP) (Roth and Yih, 2007; Chang et al., 2008). Due to space limitations, we refer interested readers to the papers for more information on the CCM framework.

By doing this, not only are the predictions of both classifiers coherent with each other (thus obtaining better predictions from both classifiers), but more importantly, we are effectively using the (more reliable) predictions of the coarse-grained classifier to constrain the predictions of the fine-grained classifier. To the best of our knowledge, this approach for RE is novel.

In this paper, we work on the NIST Automatic Content Extraction (ACE) 2004 corpus. ACE defines several coarse-grained relations such as *employment/membership*, *geo-political entity (GPE) affiliation*, etc. Each coarse-grained relation is further refined into several fine-grained relations[1] and each fine-grained relation has a unique parent coarse-grained relation. For instance, the fine-grained relations *employed as ordinary staff*, *employed as an executive*, etc. are children relations of *employment/membership*.

Let m_i and m_j denote a pair of mentions i and j drawn from a document containing N mentions. Let $R_{i,j}$ denote a relation between m_i and m_j, and let $\mathcal{R} = \{R_{i,j}\}$, where $1{\le}i, j{\le}N; i{\ne}j$ denote the set of relations in the document. Also, we denote the set of predefined coarse-grained relation types and fine-grained relation types as $\mathcal{L}_{Rc}$ and $\mathcal{L}_{Rf}$ respectively. Since there could possibly be no relation between a mention pair, we add the *null* label to $\mathcal{L}_{Rc}$ and $\mathcal{L}_{Rf}$, allowing our classifiers to predict *null* for $R_{i,j}$. Finally, for a fine-grained relation type rf, let $\mathcal{V}(rf)$ denote its parent coarse-grained relation type.

[1] With the exception of the *Discourse* coarse-grained relation.

We learn two classifiers, one for disambiguating between the coarse-grained relations and one for disambiguating between the fine-grained relations. Let θ_c and θ_f denote the feature weights learned for predicting coarse-grained and fine-grained relations respectively. Let $p_R(rc) = \log P_c(rc|m_i, m_j; \theta_c)$ be the log probability that relation R is predicted to be of coarse-grained relation type rc. Similarly, let $p_R(rf) = \log P_f(rf|m_i, m_j; \theta_f)$ be the log probability that relation R is predicted to be of fine-grained relation type rf. Let $x_{\langle R,rc\rangle}$ be a binary variable which takes on the value of 1 if relation R is labeled with the coarse-grained label rc. Similarly, let $y_{\langle R,rf\rangle}$ be a binary variable which takes on the value of 1 if relation R is labeled with the fine-grained label rf. Our objective function is then:

$$\max \sum_{R\in\mathcal{R}} \sum_{rc\in\mathcal{L}_{Rc}} p_R(rc)\cdot x_{\langle R,rc\rangle} + \sum_{R\in\mathcal{R}} \sum_{rf\in\mathcal{L}_{Rf}} p_R(rf)\cdot y_{\langle R,rf\rangle} \quad (1)$$

subject to the following constraints:

$$\sum_{rc\in\mathcal{L}_{Rc}} x_{\langle R,rc\rangle} = 1 \quad \forall R\in\mathcal{R} \quad (2)$$

$$\sum_{rf\in\mathcal{L}_{Rf}} y_{\langle R,rf\rangle} = 1 \quad \forall R\in\mathcal{R} \quad (3)$$

$$x_{\langle R,rc\rangle} \in \{0,1\} \quad \forall R\in\mathcal{R}, rc\in\mathcal{L}_{Rc} \quad (4)$$

$$y_{\langle R,rf\rangle} \in \{0,1\} \quad \forall R\in\mathcal{R}, rf\in\mathcal{L}_{Rf} \quad (5)$$

Equations (2) and (3) require that each relation can only be assigned one coarse-grained label and one fine-grained label. Equations (4) and (5) indicate that $x_{\langle R,rc\rangle}$ and $y_{\langle R,rf\rangle}$ are binary variables. Two more constraints follow:

$$x_{\langle R,rc\rangle} \leq \sum_{\{rf\in\mathcal{L}_{Rf}|\mathcal{V}(rf)=rc\}} y_{\langle R,rf\rangle} \quad \forall R\in\mathcal{R}\ , rc\in\mathcal{L}_{Rc} \quad (6)$$

$$y_{\langle R,rf\rangle} \leq x_{\langle R,\mathcal{V}(rf)\rangle} \quad \forall R\in\mathcal{R}, rf\in\mathcal{L}_{Rf} \quad (7)$$

The logical form of Equation (6) can be written as: $x_{\langle R,rc\rangle} \Rightarrow y_{\langle R,rf_1\rangle} \vee y_{\langle R,rf_2\rangle} \ldots \vee y_{\langle R,rf_n\rangle}$, where $rf_1, rf_2, \ldots, rf_n$ are (child) fine-grained relations of the coarse-grained relation rc. This states that if we assign rc to relation R, then we must also assign to R a fine-grained relation rf which is a *child* of rc. The logical form of Equation (7) can be written as: $y_{\langle R,rf\rangle} \Rightarrow x_{\langle R,\mathcal{V}(rf)\rangle}$. This captures the inverse relation and states that if we assign rf to R, then we must also assign to R the relation type $\mathcal{V}(rf)$, which is the *parent* of rf. Together, Equations (6) and (7) constrain the predictions of the coarse-grained and fine-grained classifiers to be coherent with each other. Finally, we note that one could automatically translate logical constraints into linear inequalities (Chang et al., 2008).

This method is general and is applicable to other NLP tasks where a hierarchy exists, such as WSD and question answering. For instance, in WSD, one can predict coarse-grained and fine-grained senses using suitably defined sense inventories and then perform inference via ILP to obtain coherent predictions.

art:	$E_i \in$ {gpe, org, per}, $E_j \in$ {fac, gpe, veh, wea}
emp-org:	$E_i \in$ {gpe, org, per}, $E_j \in$ {gpe, org, per}
gpe-aff:	$E_i \in$ {gpe, org, per}, $E_j \in$ {gpe, loc}
other-aff:	$E_i \in$ {gpe, org, per}, $E_j \in$ {gpe, loc}
per-soc:	$E_i \in$ {per}, $E_j \in$ {per}

Table 2: Entity type constraints.

3.2 Entity Type Constraints

Each mention in ACE-2004 is annotated with one of seven coarse-grained entity types: person (per), organization (org), location (loc), geo-political entity (gpe), facility (fac), vehicle (veh), and weapon (wea).

Roth and Yih (2007) had shown that entity type information is useful for constraining the possible labels that a relation R can assume. For instance, both mentions involved in a *personal/social* relation must be of entity type *per*. In this work, we gather such information from the ACE-2004 documentation and inject it as constraints (on the coarse-grained relations) into our system. Due to space limitations, we do not state the constraint equations or objective function here, but we list the entity type constraints we imposed for each coarse-grained relation m_i-R-m_j in Table

2^2, where E_i (E_j) denotes the allowed set of entity types for mention m_i (m_j). Applying the entity type information improves the predictions of the coarse-grained classifier and this in turn could improve the predictions of the fine-grained classifier.

3.3 Using Coreference Information

We can also utilize the coreference relations among entity mentions. Assuming that we know mentions m_i and m_j are coreferent with each other, then there should be no relation between them[3]. Let $z_{\langle i,j\rangle}$ be a binary variable which takes on the value of 1 if mentions m_i and m_j are coreferent, and 0 if they are not. When $z_{\langle i,j\rangle}$=1, we capture the above intuition with the following constraints:

$$z_{\langle i,j\rangle} \leq x_{\langle R_{i,j}, null\rangle} \quad (8)$$

$$z_{\langle i,j\rangle} \leq y_{\langle R_{i,j}, null\rangle} \quad (9)$$

which can be written in logical form as: $z_{\langle i,j\rangle} \Rightarrow x_{\langle R_{i,j}, null\rangle}$, and $z_{\langle i,j\rangle} \Rightarrow y_{\langle R_{i,j}, null\rangle}$. We add the following to our objective function in Equation (1):

$$\sum_{m_i, m_j \in \mathbf{m}^2} co_{\langle i,j\rangle} \cdot z_{\langle i,j\rangle} + \bar{co}_{\langle i,j\rangle} \cdot (1 - z_{\langle i,j\rangle}) \quad (10)$$

where **m** is the set of mentions in a document, $co_{\langle i,j\rangle}$ and $\bar{co}_{\langle i,j\rangle}$ are the log probabilities of predicting that m_i and m_j are coreferent and not coreferent respectively. In this work, we assume we are given coreference information, which is available from the ACE annotation.

3.4 Using Knowledge from Wikipedia

We propose two ways of using Wikipedia to gather features for relation extraction. Wikipedia is a huge online encyclopedia and mainly contains articles describing entities or concepts.

The first intuition is that if we are able to correctly map a pair of mentions m_i and m_j to their corresponding Wikipedia article (assuming they are represented in Wikipedia), we could use the content on their Wikipedia pages to check whether they are related.

In this work, we use a *Wiki* system (Ratinov et al., 2010) which performs context-sensitive mapping of mentions to Wikipedia pages. In their work, the authors first identify phrases or mentions that could be mapped. The correct Wikipedia article for each mention is then probabilistically predicted using a combination of features based on Wikipedia hyperlink structure, semantic coherence, etc. The authors' own evaluation results indicate that the performance of their system ranges from 70–80%. When given a pair of mentions and the system returns the Wikipedia page for either one of the mentions, we introduce a feature:

$$w_1(m_i, m_j) = \begin{cases} 1, & \text{if } A_{m_i}(m_j) \\ & \text{or } A_{m_j}(m_i) \\ 0, & \text{otherwise} \end{cases}$$

where $A_{m_i}(m_j)$ returns true if the head extent of m_j is found (via simple string matching) in the predicted Wikipedia article of m_i. The interpretation of $A_{m_j}(m_i)$ is similar. We introduce a new feature into the RE system by combining $w_1(m_i, m_j)$ with m_i, m_j E-maintype (defined as in Table 1).

The second feature based on Wikipedia is as follows. It will be useful to check whether there is any *parent-child* relationship between two mentions. Intuitively, this will be useful for recognizing several relations such as *physical part-whole* (e.g. a city is part of a state), *subsidiary* (a company is a child-company of another), *citizenship* (a person is a citizen of a country), etc.

Given a pair of mentions m_i and m_j, we use a *Parent-Child* system (Do and Roth, 2010) to predict whether they have a *parent-child* relation. To achieve this, the system first gathers all Wikipedia articles that are related to m_i and m_j. It then uses the words in these pages and the category ontology of Wikipedia to make its *parent-child* predictions, while respecting certain defined constraints. In this work, we use its prediction as follows:

$$w_2(m_i, m_j) = \begin{cases} 1, & \text{if } parent\text{-}child(m_i, m_j) \\ 0, & \text{otherwise} \end{cases}$$

[2] We do not impose entity type constraints on the coarse-grained relations *disc* and *phys*.

[3] In this work, we assume that no relations are reflexive. After the experiments in this paper are performed, we verified that in the ACE corpus we used, less than 1% of the relations are reflexive.

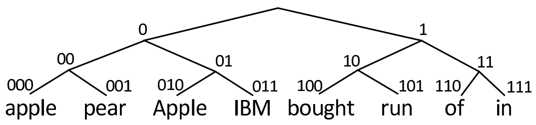


Figure 1: An example of Brown word cluster hierarchy from (Koo et al., 2008).

where we combine $w_2(m_i, m_j)$ with m_i, m_j E-maintype, introducing this as a new feature into our RE system.

3.5 Using Word Clusters

An inherent problem faced by supervised systems is that of data sparseness. To mitigate such issues in the lexical features, we use word clusters which are automatically generated from unlabeled texts. In this work, we use the Brown clustering algorithm (Brown et al., 1992), which has been shown to improve performance in various NLP applications such as dependency parsing (Koo et al., 2008), named entity recognition (Ratinov and Roth, 2009), and relation extraction (Boschee et al., 2005). The algorithm performs a hierarchical clustering of the words and represents them as a binary tree.

Each word is uniquely identified by its path from the root and every path is represented with a bit string. Figure 1 shows an example clustering where the maximum path length is 3. By using path prefixes of different lengths, one can obtain clusterings at different granularity. For instance, using prefixes of length 2 will put *apple* and *pear* into the same cluster, *Apple* and *IBM* into the same cluster, etc. In our work, we use clusters generated from New York Times text and simply use a path prefix of length 10. When Brown clusters are used in our system, all lexical features consisting of single words will be duplicated. For instance, for the feature *hw of m1*, one new feature which is the length-10 bit string path representing the original lexical head word of m1, will be introduced and presented to the classifier as a string feature.

4 Experiments

We used the ACE-2004 dataset (catalog LDC2005T09 from the Linguistic Data Consortium) to conduct our experiments. ACE-2004 defines 7 coarse-grained relations and 23 fine-grained relations. In all of our experiments, unless otherwise stated, we explicitly model the argument order (of the mentions) when asked to disambiguate the relation between a pair of mentions. Hence, we built our coarse-grained classifier with 15 relation labels to disambiguate between (two for each coarse-grained relation type and a *null* label when the two mentions are not related). Likewise, our fine-grained classifier has to disambiguate between 47 relation labels. In the dataset, relations do not cross sentence boundaries.

For our experiments, we trained regularized averaged perceptrons (Freund and Schapire, 1999), implemented within the Sparse Network of Winnow framework (Carlson et al., 1999), one for predicting the coarse-grained relations and another for predicting the fine-grained relations. Since the dataset has no split of training, development, and test sets, we followed prior work (Jiang and Zhai, 2007) and performed 5-fold cross validation to obtain our performance results. For simplicity, we used 5 rounds of training and a regularization parameter of 1.5 for the perceptrons in all our experiments. Finally, we concentrate on the evaluation of fine-grained relations.

4.1 Performance of the Basic RE system

As a gauge on the performance of our basic relation extraction system BasicRE using only the features described in Section 2, we compare against the state-of-the-art feature-based RE system of Jiang and Zhai (2007). However, we note that in that work, the authors performed their evaluation using *undirected* coarse-grained relations. That is, they do not distinguish on argument order of mentions and the classifier has to decide among 8 relation labels (7 coarse-grained relation types and a *null* label). Performing 5-fold cross validation on the news wire (nwire) and broadcast news (bnews) corpora in the ACE-2004 dataset, they reported a F-measure of 71.5 using a maximum entropy classifier[4]. Evaluating BasicRE on the same setting,

[4]After they heuristically performed feature selection and applied the heuristics giving the best evaluation performance, they obtained a result of 72.9.

	All nwire			10% of nwire		
Features	Rec%	Pre%	F1%	Rec%	Pre%	F1%
BasicRE	49.9	51.0	50.5	33.2	29.0	31.0
+Hier	+1.3	+1.3	+1.3	+1.1	+1.2	+1.1
+Hier+relEntC	+1.5	+2.0	+1.8	+3.3	+3.5	+3.4
+Coref	$\sim$	+1.4	+0.7	−0.1	+1.0	+0.5
+Wiki	+0.2	+1.9	+1.0	+1.5	+2.5	+2.0
+Cluster	−0.2	+3.2	+1.4	−0.7	+3.9	+1.7
+ALL	+1.5	+6.7	+3.9	+4.7	+10.2	+7.6

Table 3: BasicRE gives the performance of our basic RE system on predicting fine-grained relations, obtained by performing 5-fold cross validation on only the news wire corpus of ACE-2004. Each subsequent row +Hier, +Hier+relEntC, +Coref, +Wiki, and +Cluster gives the *individual* contribution from using each knowledge. The bottom row +ALL gives the performance improvements from adding +Hier+relEntC+Coref+Wiki+Cluster. $\sim$ indicates no change in score.

we obtained a competitive F-measure of 71.2[5].

4.2 Experimental Settings for Evaluating Fine-grained Relations

Two of our knowledge sources, the Wiki system described in Section 3.4 and the word clusters described in Section 3.5, assume inputs of mixed-cased text. We note that the bnews corpus of ACE-2004 is entirely in lower-cased text. Hence, we use only the nwire corpus for our experiments here, from which we gathered 28,943 relation instances and 2,226 of those have a valid (non-null) relation[6].

We also propose the following experimental setting. First, since we made use of coreference information, we made sure that while performing our experiments, all instances from the same document are either all used as training data or all used as test data. Prior work in RE had not ensured this, but we argue that this provides a more realistic setting. Our own experiments indicate that this results in a 1-2% lower performance on fine-grained relations.

Secondly, prior work calculate their performance on relation extraction at the level of *mentions*. That is, each mention pair extracted is scored individually. An issue with this way of scoring on the ACE corpus is that ACE annotators rarely duplicate a relation link for coreferent mentions. For instance, assume that mentions m_i, m_j, and m_k exist in a given sentence, mentions m_i and m_j are coreferent, and the annotator establishes a particular relation type r between m_j and m_k. The annotator will not usually duplicate the same relation r between m_i and m_k and thus the label between these two mentions is then *null*. We are not suggesting that this is an incorrect approach, but clearly there is an issue since an important goal of performing RE is to populate or build an ontology of entities and establish the relations existing among the entities. Thus, we evaluate our performance at the *entity-level*.[7] That is, given a pair of entities, we establish the set of relation types existing between them, based on their mention annotations. Then we calculate recall and precision based on these established relations. Of course, performing such an evaluation requires knowledge about the coreference relations and in this work, we assume we are given this information.

4.3 Knowledge-Enriched System

Evaluating our system BasicRE (trained only on the features described in Section 2) on the nwire corpus, we obtained a F1 score of 50.5, as shown in Table 3. Next, we exploited the relation hierarchy as in Section 3.1 and obtained an improvement of 1.3, as shown in the row +Hier. Next, we added the entity type constraints of Section

[5]Using 10 rounds of training and a regularization parameter of 2.5 improves the result to 72.2. In general, we found that more rounds of training and a higher regularization value benefits coarse-grained relation classification, but not fine-grained relation classification.

[6]The number of relation instances in the nwire and bnews corpora are about the same.

[7]Our experiments indicate that performing the usual evaluation on mentions gives similar performance figures and the trend in Table 3 stays the same.

3.2. Remember that these constraints are imposed on the coarse-grained relations. Thus, they would only affect the fine-grained relation predictions if we also exploit the relation hierarchy. In the table, we show that all the background knowledge helped to improve performance, providing a total improvement of 3.9 to our basic RE system. Though the focus of this work is on fine-grained relations, our approach also improves the performance of coarse-grained relation predictions. BasicRE obtains a F1 score of 65.3 on coarse-grained relations and exploiting background knowledge gives a total improvement of 2.9.

5 Analysis

We explore the situation where we have very little training data. We assume during each cross validation fold, we are given only 10% of the training data we originally had. Previously, when performing 5-fold cross validation on 2,226 valid relation instances, we had about 1780 as training instances in each fold. Now, we assume we are only given about 178 training instances in each fold. Under this condition, BasicRE gives a F1 score of 31.0 on fine-grained relations. Adding all the background knowledge gives an improvement of 7.6 and this represents an error reduction of 39% when measured against the performance difference of 50.5 (31.0) when we have 1780 training instances vs. 178 training instances. On the coarse-grained relations, BasicRE gives a F1 score of 51.1 and exploiting background knowledge gives a total improvement of 5.0.

We also tabulated the list of fine-grained relations that improved by more than 1 F1 score when we incorporated +Wiki, on the experiment using all of nwire data: *phys:near* (physically near), *other-aff:ideology* (ideology affiliation), *art:user-or-owner* (user or owner of artifact), *per-soc:business* (business relationship), *phys:part-whole* (physical part-whole), *emp-org:subsidiary* (organization subsidiary), and *gpe-aff:citizen-or-resident* (citizen or resident). Most of these intuitively seemed to be information one would find being mentioned in an encyclopedia.

6 Related Work

Few prior work has explored using background knowledge to improve relation extraction performance. Zhou et al. (2008) took advantage of the hierarchical ontology of relations by proposing methods customized for the perceptron learning algorithm and support vector machines. In contrast, we propose a generic way of using the relation hierarchy which at the same time, gives globally coherent predictions and allows for easy injection of knowledge as constraints. Recently, Jiang (2009) proposed using features which are common across all relations. Her method is complementary to our approach, as she does not consider information such as the relatedness between different relations. On using semantic resources, Zhou et al. (2005) gathered two gazettes, one containing country names and another containing words indicating personal relationships. In relating the tasks of RE and coreference resolution, Ji et al. (2005) used the output of a RE system to rescore coreference hypotheses. In our work, we reverse the setting and explore using coreference to improve RE.

7 Conclusion

In this paper, we proposed a broad range of methods to inject background knowledge into a relation extraction system. Some of these methods, such as exploiting the relation hierarchy, are general in nature and could be easily applied to other NLP tasks. To combine the various relation predictions and knowledge, we perform global inference within an ILP framework. Besides allowing for easy injection of knowledge as constraints, this ensures globally coherent models and predictions.

Acknowledgements This research was partly sponsored by Air Force Research Laboratory (AFRL) under prime contract no. FA8750-09-C-0181. We thank Ming-Wei Chang and James Clarke for discussions on this research.

References

Banko, Michele, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007.

Open information extraction from the web. In *Proceedings of IJCAI-07*, pages 2670–2676.

Boschee, Elizabeth, Ralph Weischedel, and Alex Zamanian. 2005. Automatic information extraction. In *Proceedings of the International Conference on Intelligence Analysis*.

Brown, Peter F., Vincent J. Della Pietra, Peter V. deSouza, Jenifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479.

Bunescu, Razvan C. and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of HLT/EMNLP-05*, pages 724–731.

Carlson, Andrew J., Chad M. Cumby, Jeff L. Rosen, and Dan Roth. 1999. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department, May.

Chang, Ming-Wei, Lev Ratinov, Nicholas Rizzolo, and Dan Roth. 2008. Learning and inference with constraints. In *Proceedings of AAAI-08*, pages 1513–1518.

Do, Quang and Dan Roth. 2010. On-the-fly constraint-based taxonomic relation identification. Technical report, University of Illinois. http://L2R.cs.uiuc.edu/∼danr/Papers/DoRo10.pdf.

Freund, Yoav and Robert E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296.

Ji, Heng, David Westbrook, and Ralph Grishman. 2005. Using semantic relations to refine coreference decisions. In *Proceedings of HLT/EMNLP-05*, pages 17–24.

Jiang, Jing and ChengXiang Zhai. 2007. A systematic exploration of the feature space for relation extraction. In *Proceedings of HLT-NAACL-07*, pages 113–120.

Jiang, Jing. 2009. Multi-task transfer learning for weakly-supervised relation extraction. In *Proceedings of ACL-IJCNLP-09*, pages 1012–1020.

Kambhatla, Nanda. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *Proceedings of ACL-04*, pages 178–181.

Koo, Terry, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08:HLT*, pages 595–603.

Mintz, Mike, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL-IJCNLP-09*, pages 1003–1011.

Ratinov, Lev and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL-09*, pages 147–155.

Ratinov, Lev, Doug Downey, and Dan Roth. 2010. Wikification for information retrieval. Technical report, University of Illinois. http://L2R.cs.uiuc.edu/∼danr/Papers/RatinovDoRo10.pdf.

Roth, Dan and Wen Tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of CoNLL-04*, pages 1–8.

Roth, Dan and Wen Tau Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. In Getoor, Lise and Ben Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press.

Weld, Daniel S., Raphael Hoffman, and Fei Wu. 2008. Using wikipedia to bootstrap open information extraction. *ACM SIGMOD Special Issue on Managing Information Extraction*, 37(4):62–68.

Zelenko, Dmitry, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

Zhang, Min, Jie Zhang, Jian Su, and GuoDong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of COLING-ACL-06*, pages 825–832.

Zhou, Guodong, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of ACL-05*.

Zhou, GuoDong, Min Zhang, DongHong Ji, and QiaoMing Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *Proceedings of EMNLP-CoNLL-07*, pages 728–736.

Zhou, Guodong, Min Zhang, Dong-Hong Ji, and Qiaoming Zhu. 2008. Hierarchical learning strategy in semantic relation extraction. *Information Processing & Management*, 44(3):1008–1021.

Jointly Modeling WSD and SRL with Markov Logic

Wanxiang Che and Ting Liu
Research Center for Information Retrieval
MOE-Microsoft Key Laboratory of Natural Language Processing and Speech
School of Computer Science and Technology
{car, tliu}@ir.hit.edu.cn

Abstract

Semantic role labeling (SRL) and word sense disambiguation (WSD) are two fundamental tasks in natural language processing to find a sentence-level semantic representation. To date, they have mostly been modeled in isolation. However, this approach neglects logical constraints between them. We therefore exploit some pipeline systems which verify the automatic all word sense disambiguation could help the semantic role labeling and vice versa. We further propose a Markov logic model that jointly labels semantic roles and disambiguates all word senses. By evaluating our model on the OntoNotes 3.0 data, we show that this joint approach leads to a higher performance for word sense disambiguation and semantic role labeling than those pipeline approaches.

1 Introduction

Semantic role labeling (SRL) and word sense disambiguation (WSD) are two fundamental tasks in natural language processing to find a sentence-level semantic representation. Semantic role labeling aims at identifying the relations between predicates in a sentence and their associated arguments. Word sense disambiguation is the process of identifying the correct meaning, or sense of a word in a given context. For example, for the sentence in Figure 1, we can find out that the predicate token "hitting" at position 3 has sense "cause to move by striking" and the sense label is "hit.01". The argument headed by the token "cat" at position 1 with sense "feline mammal" (cat.01) is referring to the player (A0), and the argument headed by the token "ball" at position 5 with sense

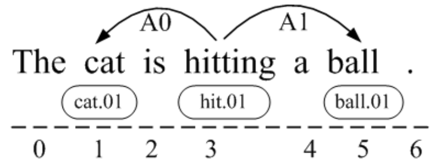


Figure 1: A sample of word sense disambiguation and semantic role labeling.

"round object that is hit in games" (ball.01) is referring to the game object (A1) being hit.

Normally, semantic role labeling and word sense disambiguation are regarded as two independent tasks, i.e., the word sense information is rarely used in a semantic role labeling system and vice versa. A few researchers have used semantic roles to help the verb sense disambiguation (Dang and Palmer, 2005). More people used predicate senses in semantic role labeling (Hajič et al., 2009; Surdeanu et al., 2008). However, both of the pipeline methods ignore possible dependencies between the word senses and semantic roles, and can result in the error propagation problem. The same problem also appears in other natural language processing tasks.

In order to make different natural language processing tasks be able to help each other, jointly modeling methods become popular recently, such as joint Chinese word segmentation and part-of-speech tagging (Kruengkrai et al., 2009; Zhang and Clark, 2008; Jiang et al., 2008), joint lemmatization and part-of-speech prediction (Toutanova and Cherry, 2009), joint morphological segmentation and syntactic parsing (Goldberg and Tsarfaty, 2008), joint text and aspect ratings for sentiment summarization (Titov and McDonald, 2008), and joint parsing and named entity recognition (Finkel and Manning, 2009). For semantic role labeling, Dahlmeier et al. (2009) proposed a method to maximize the joint probability of the seman-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 161–169,
Beijing, August 2010

tic role of preposition phrases and the preposition sense.

In order to do better joint learning, a novel statistical relational learning framework, Markov logic (Domingos and Lowd, 2009) was introduced to join semantic role labeling and predicate senses (Meza-Ruiz and Riedel, 2009). Markov logic combines the first order logic and Markov networks, to develop a joint probability model over all related rules. Global constraints (introduced by Punyakanok et al. (2008)) among semantic roles can be easily added into Markov logic. And the more important, the jointly modeling can be realized using Markov logic naturally.

Besides predicates and prepositions, other word senses are also important information for recognizing semantic roles. For example, if we know "cat" is an "agent" of the predicate "hit" in a sentence, we can guess that "dog" can also be an "agent" of "hit", though it does not appear in the training data. Similarly, the semantic role information can also help to disambiguate word senses. In addition, the predicate sense and the argument sense can also help each other. In the sentence "The cat is hitting a ball.", if we know "hit" here has a game related sense, we can guess that the "ball" should have the sense "is a round object in games". In the same way, the correct "ball" sense can help to disambiguate the sense of "hit". The joint probability, that they are disambiguated correctly simultaneously will be larger than other abnormalities.

The release of OntoNotes (Hovy et al., 2006) provides us an opportunity to jointly model all word senses disambiguation and semantic role labeling. OntoNotes is a large corpus annotated with constituency trees (based on Penn Treebank), predicate argument structures (based on Penn PropBank), all word senses, etc. It has been used in some natural language processing tasks, such as joint parsing and named entity recognition (Finkel and Manning, 2009), and word sense disambiguation (Zhong et al., 2008).

In this paper, we first propose some pipeline systems which exploit automatic all word sense disambiguation into semantic role labeling task and vice versa. Then we present a Markov logic model which can easily express useful global constraints and jointly disambiguate all word senses and label semantic roles.

Experiments on the OntoNotes 3.0 corpus show that (1) the automatic all word sense disambiguation and semantic role labeling tasks can help each other when using pipeline approaches, and more important, (2) the joint approach using Markov logic leads to higher accuracy for word sense disambiguation and performance (F_1) for semantic role labeling than pipeline approaches.

2 Related Work

Joint models were often used in semantic role labeling community. Toutanova et al. (2008) and Punyakanok et al. (2008) presented a re-ranking model and an integer linear programming model respectively to jointly learn a global optimal semantic roles assignment. Besides jointly learning semantic role assignment of different constituents for one task (semantic role labeling), their methods have been used to jointly learn for two tasks (semantic role labeling and syntactic parsing). However, it is easy for the re-ranking model to loss the optimal result, if it is not included in the top n results. In addition, the integer linear programming model can only use hard constraints. A lot of engineering work is also required in both models.

Recently, Markov logic (Domingos and Lowd, 2009) became a hot framework for joint model. It has been successfully used in temporal relations recognition (Yoshikawa et al., 2009), co-reference resolution (Poon and Domingos, 2008), etc. It is very easy to do joint modeling using Markov logic. The only work is to define relevant formulas. Meza-Ruiz and Riedel (2009) have joined semantic role labeling and predicate senses disambiguation with Markov logic.

The above idea, that the predicate senses and the semantic role labeling can help each other, may be inspired by Hajič et al. (2009), Surdeanu et al. (2008), and Dang and Palmer (2005). They have shown that semantic role features are helpful to disambiguate verb senses and vice versa.

Besides predicate senses, Dahlmeier et al. (2009) proposed a joint model to maximize probability of the preposition senses and the semantic role of prepositional phrases.

Except for predicate and preposition senses, Che et al. (2010) explored all word senses for semantic role labeling. They showed that all word senses can improve the semantic role labeling performance significantly. However, the golden word senses were used in their experiments. The results are still unknown when an automatic word sense disambiguation system is used.

In this paper, we not only use all word senses disambiguated by an automatic system, but also make the semantic role labeling results to help word sense disambiguation synchronously with a joint model.

3 Markov Logic

Markov logic can be understood as a knowledge representation with a weight attached to a first-order logic formula. Let us describe Markov logic in the case of the semantic role labeling task. We can model this task by first introducing a set of logical predicates such as $role(p, a, r)$ and $lemma(i, l)$, which means that the argument at position a has the role r with respect to the predicate at position p and token at position i has lemma l respectively. Then we specify a set of weighted first order formulas that define a distribution over sets of ground atoms of these predicates (or so-called possible worlds).

Ideally, the distribution we define with these weighted formulas assigns high probability to possible worlds where semantic role labeling is correct and a low probability to worlds where this is not the case. For instance, for the sentence in Figure 1, a suitable set of weighted formulas would assign a high probability to the world:

$$lemma(1, cat), lemma(3, hit), lemma(5, ball)$$
$$role(3, 1, A0), role(3, 5, A1)$$

and low probabilities to other cases.

A Markov logic network (MLN) M is a set of weighted formulas, i.e., a set of pairs (ϕ, ω), where ϕ is a first order formula and ω is the real weight of the formula. M defines a probability distribution over possible worlds:

$$p(y) = \frac{1}{Z} \exp\Big(\sum_{(\phi,\omega)\in M} \omega \sum_{c \in C^{\phi}} f_c^{\phi}(y)\Big)$$

where each c is a binding of free variables in ϕ to constants. Each f_c^{ϕ} is a binary feature function that returns 1 if the possible world y includes the ground formula by replacing the free variables in ϕ with the constants in c is true, and 0 otherwise. C^{ϕ} is the set of all bindings for the variables in ϕ. Z is a normalization constant.

4 Model

We divide our system into two stages: word sense disambiguation and semantic role labeling. For comparison, we can process them with pipeline strategy, i.e., the word sense disambiguation results are used in semantic role labeling or the semantic role labeling results are used in word sense disambiguation. Of course, we can jointly process them with Markov logic easily.

We define two hidden predicates for the two stages respectively. For word sense disambiguation, we define the predicate $sense(w, s)$ which indicates that the word at position w has the sense s. For semantic role labeling, the predicate $role(p, a, r)$ is defined as mentioned in above.

Different from Meza-Ruiz and Riedel (2009), which only used sense number as word sense representation, we use a triple (lemma, part-of-speech, sense num) to represent the word sense s. For example, (hit, v, 01) denotes that the verb "hit" has sense number 01. Obviously, our representation can distinguish different word senses which have the identical sense number. In addition, we use one argument classification stage with predicate $role$ to label semantic roles as Che et al. (2009). Similarly, no argument identification stage is used in our model. The approach can improve the recall of the system.

In addition to the hidden predicates, we define observable predicates to represent the information available in the corpus. Table 1 presents these predicates.

4.1 Local Formula

A local formula means that its groundings relate any number of observed ground atoms to exactly one hidden ground atom. For example

$$lemma(p, +l_1) \wedge lemma(a, +l_2) \Rightarrow role(p, a, +r)$$

Predicates	Description
$word(i, w)$	Token i has word w
$pos(i, t)$	Token i has part-of-speech t
$lemma(i, l)$	Token i has lemma l
$chdpos(i, t)$	The part-of-speech string of token i's all children is t
$chddep(i, d)$	The dependency relation string of token i's all children is t
$firstLemma(i, l)$	The leftmost lemma of a subtree rooted by token i is l
$lastLemma(i, l)$	The rightmost lemma of a subtree rooted by token i is l
$posFrame(i, fr)$	fr is a part-of-speech frame at token i
$dep(h, a, de)$	The dependency relation between an argument a and its head h is de
$isPredicate(p)$	Token p is a predicate
$posPath(p, a, pa)$	The part-of-speech path between a predicate p and an argument a is pa
$depPath(p, a, pa)$	The dependency relation path between a predicate p and an argument a is pa
$pathLen(p, a, le)$	The path length between a predicate p and an argument a is le
$position(p, a, po)$	The relative position between a predicate p and an argument a is po
$family(p, a, fa)$	The family relation between a predicate p and an argument a is fa
$wsdCand(i, t)$	Token i is a word sense disambiguation candidate, here t is "v" or "n"
$uniqe(r)$	For a predicate, semantic role r can only appear once

Table 1: Observable Predicates.

means that if the predicate lemma at position p is l_1 and the argument lemma at position a is l_2, then the semantic role between the predicate and the argument is r with some possibility.

The $+$ notation signifies that Markov logic generates a separate formula and a separate weight for each constant of the appropriate type, such as each possible pair of lemmas (l_1, l_2, r). This type of "template-based" formula generation can be performed automatically by a Markov logic engine, such as the *thebeast*[1] system.

The local formulas are based on features employed in the state-of-the-art systems. For word sense disambiguation, we use the basic features mentioned by Zhong et al. (2008). The semantic role labeling features are from Che et al. (2009), the best system of the CoNLL 2009 shared task. The final features are listed in Table 2.

[1]http://code.google.com/p/thebeast/

Features	SRL	WSD
Lemma	•	•
POS	•	•
FirstwordLemma	•	
HeadwordLemma	•	
HeadwordPOS	•	
LastwordLemma	•	
POSPath	•	
PathLength	•	
Position	•	
PredicateLemma	•	
PredicatePOS	•	
RelationPath	•	
DepRelation	•	
POSUpPath	•	
POSFrame	•	
FamilyShip	•	
BagOfWords		•
Window3OrderedWords		•
Window3OrderedPOSs		•

Table 2: Local Features.

What follows are some simple examples in order to explain how we implement each feature as a formula (or a set of formulas).

Consider the "Position" feature. We first introduce a predicate $position(p, a, po)$ that denotes the relative position between predicate p and argument a is po. Then we add a formula

$$position(p, a, +po) \Rightarrow role(p, a, +r)$$

for all possible combinations of $position$ and $role$ relations.

The "BagOfWords" feature means that the sense of a word w is determined by all of lemmas in a sentence. Then, we add the following formula set:

$$wsdCand(w, +t_w) \wedge lemma(w, +l_w) \wedge lemma(1, +l_1) \Rightarrow sense(w, +s)$$
$$\ldots$$
$$wsdCand(w, +t_w) \wedge lemma(w, +l_w) \wedge lemma(2, +l_i) \Rightarrow sense(w, +s)$$
$$\ldots$$
$$wsdCand(w, +t_w) \wedge lemma(w, +l_w) \wedge lemma(n, +l_n) \Rightarrow sense(w, +s)$$

where, the w is the position of current word and t_w is its part-of-speech tag, l_w is its lemma. l_i is the lemma of token i. There are n tokens in a sentence totally.

4.2 Global Formula

Global formulas relate more than one hidden ground atoms. We use this type of formula for two purposes:

1. To capture the global constraints among different semantic roles;

2. To reflect the joint relation between word sense disambiguation and semantic role labeling.

Punyakanok et al. (2008) proposed an integer linear programming (ILP) model to get the global optimization for semantic role labeling, which satisfies some constraints. This approach has been successfully transferred into dependency parse tree based semantic role labeling system by Che et al. (2009). The final results must satisfy two constraints which can be described with Markov logic formulas as follows:

C1: Each word should be labeled with one and only one label.

$$role(p, a, r_1) \land r_1 \neq r_2 \Rightarrow \neg role(p, a, r_2)$$

The same unique constraint also happens on the word sense disambiguation, i.e.,

$$sense(w, s_1) \land s_1 \neq s_2 \Rightarrow \neg sense(w, s_2)$$

C2: Some roles (A0∼A5) appear only once for a predicate.

$$role(p, a_1, r) \land uniqe(r) \land a_1 \neq a_2 \Rightarrow \neg role(p, a_2, r)$$

It is also easy to express the joint relation between word sense disambiguation and semantic role labeling with Markov logic. What we need to do is just adding some global formulas. The relation between them can be shown in Figure 2. Inspired by CoNLL 2008 (Surdeanu et al., 2008) and 2009 (Hajič et al., 2009) shared tasks, where most of successful participant systems used predicate senses for semantic role labeling, we also model that the word sense disambiguation implicates the semantic role labeling.

Here, we divide the all word sense disambiguation task into two subtasks: predicate sense disambiguation and argument sense disambiguation. The advantages of the division method approach lie in two aspects. First, it makes us distinguish the contributions of predicate and argument word sense disambiguation respectively. Second, as previous discussed, the predicate and argument sense disambiguation can help each other. Therefore, we can reflect the help with the division and use Markov logic to represent it.

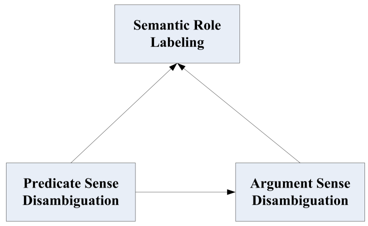


Figure 2: Global model between word sense disambiguation and semantic role labeling.

Finally, we use three global formulas to implement the three lines with direction in Figure 2. They are:

$$\begin{aligned} sense(p, +s) &\Rightarrow role(p, a, +r) \\ sense(a, +s) &\Rightarrow role(p, a, +r) \\ sense(p, +s) &\Rightarrow sense(a, +s) \end{aligned}$$

5 Experiments

5.1 Experimental Setting

In our experiments, we use the OntoNotes Release 3.0[2] corpus, the latest version of OntoNotes (Hovy et al., 2006). The OntoNotes project leaders describe it as “a large, multilingual richly-annotated corpus constructed at 90% internanotator agreement.” The corpus has been annotated with multiple levels of annotation, including constituency trees, predicate argument structure, word senses, co-reference, and named entities. For this work, we focus on the constituency trees, word senses, and predicate argument structures. The corpus has English, Chinese, and Arabic portions, and we just use the English portion, which has been split into four sections: broadcast conversation (bc), broadcast news (bn), magazine (mz), and newswire (nw). There are several datasets in each section, such as cnn and voa.

We will do our experiments on all of the OntoNotes 3.0 English datasets. For each dataset, we aimed for roughly a 60% train / 20% development / 20% test split. See Table 3 for the detailed statistics. Here, we use the human annotated part-of-speech and parse trees provided by OntoNotes. The lemma of each word is extracted using WordNet tool[3].

[2] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2009T24

[3] http://wordnet.princeton.edu/

		Training		Developing		Testing	
bc	cctv	1,042	(0000-0003)	328	(0004-0004)	333	(0005-0005)
	cnn	2,927	(0000-0004)	963	(0005-0006)	880	(0007-0008)
	msnbc	2,472	(0000-0003)	1,209	(0004-0005)	1,315	(0006-0007)
	phoenix	590	(0000-0001)	240	(0002-0002)	322	(0003-0003)
bn	abc	594	(0001-0040)	146	(0041-0054)	126	(0057-0069)
	cnn	1,610	(0001-0234)	835	(0235-0329)	1,068	(0330-0437)
	mnb	309	(0001-0015)	111	(0016-0020)	114	(0021-0025)
	nbc	281	(0001-0023)	128	(0024-0031)	78	(0032-0039)
	pri	1,104	(0001-0068)	399	(0069-0090)	366	(0091-0112)
	voa	1,159	(0001-0159)	315	(0160-0212)	315	(0213-0265)
mz	sinorama	5,051	(1001-1048)	1,262	(1049-1063)	1,456	(1064-1078)
nw	wsj	8,138	(0020-1446)	2,549	(1447-1705)	3,133	(1730-2454)
	xinhua	2,285	(0001-0195)	724	(0196-0260)	670	(0261-0325)
All		27,562		9,209		10,176	

Table 3: Training, developing and testing set sizes for the datasets in sentences. The file ranges (in parenthesis) refer to the numbers within the names of the original OntoNotes 3.0 files. Here, we remove 4,873 sentences without semantic role labeling annotation.

Because we used semantic role labeling system which is based on dependence syntactic trees, we convert the constituency trees into dependence trees with an Constituent-to-Dependency Conversion Tool[4].

The *thebeast* system is used in our experiment as Markov logic engine. It uses cutting planes inference technique (Riedel, 2008) with integer linear programming. The weights are learned with MIRA (Crammer and Singer, 2003) online learning algorithm.

To our knowledge, this is the first word sense disambiguation and semantic role labeling experiment on OntoNotes 3.0 corpus. In order to compare our joint model with previous work, we build several systems:

Baseline: There are two independent baseline systems: word sense disambiguation and semantic role labeling. In each of baseline systems, we only use the local formulas (Section 4.1) and the global formulas which only express the global constraints (Section 4.2).

Pipeline: In a pipeline system, we use additional features outputted by preceded stages. Such as in semantic role labeling pipeline system, we use word sense as features, i.e., we set $sense(w, s)$ as an observable predicate and add $sense(p, s) \Rightarrow role(p, a, r)$ and $sense(a, s) \Rightarrow role(p, a, r)$ formulas into semantic role labeling task. As for word sense disambiguation task, we add a set of formulas $role(p, a_i, r) \Rightarrow sense(p, s)$, where a_i is the i^{th} argument of the predicate at position p, and a formula $role(p, a, r) \Rightarrow sense(p, s)$ for the argument at position a respectively.

Jointly: We use all global formulas mentioned in Section 4.2. With Markov logic, we can add global constraints and get the word sense disambiguation and the semantic role labeling results simultaneously.

5.2 Results and Discussion

The performance of these systems on test set is shown in Table 4. All of the parameters are fine tuned on the development set.

Here, we only consider the noun and verb word sense disambiguation, which cover most of multi-sense words. Therefore, the word sense disambiguation performance means the accuracy of all nouns and verbs in the test set. The performance of semantic role labeling is calculated using the semantic evaluation metric of the CoNLL 2009 shared task scorer[5]. It measures the precision, recall, and F_1 score of the recovered semantic dependencies. The F_1 score is used as the final performance metric. A semantic dependency is created for each predicate and its arguments. The label of such dependency is the role of the argument. The same with the CoNLL 2009 shared task, we assume that the predicates have been identified

[4]http://nlp.cs.lth.se/software/treebank_converter/

[5]http://ufal.mff.cuni.cz/conll2009-st/eval09.pl

		WSD	SRL
Most Frequent Sense		85.58	—
Baseline		89.37	83.97
Pipeline	PS	89.53	84.17
	AS	89.41	83.94
	PS + AS	—	84.24
Jointly	PS ⇒ SRL	89.53	84.27
	AS ⇒ SRL	89.49	84.16
	PS ⇒ AS	89.45	—
	PS + AS ⇒ SRL	89.54	84.34
	Fully	89.55	84.36

Table 4: The results of different systems. Here, PS means predicate senses and AS means argument senses.

correctly.

The first row of Table 4 gives the word sense disambiguation result with the most frequent sense, i.e., the #01 sense of each candidate word which normally is the most frequent one in a balance corpus.

The second row shows the baseline performances. Here, we note that the 89.37 word sense disambiguation accuracy and the 83.97 semantic role labeling F_1 we obtained are comparable to the state-of-the-art systems, such as the 89.1 word sense disambiguation accuracy given by Zhong et al. (2008) and 85.48 semantic role labeling performance given by Che et al. (2010) on OntoNotes 2.0 respectively, although the corpus used in our experiments is upgraded version of theirs[6]. Additionally, the performance of word sense disambiguation is higher than that of the most frequent sense significantly (z-test[7] with $\rho < 0.01$). Therefore, the experimental results show that the Markov logic can achieve considerable performances for word sense disambiguation and semantic role labeling on the latest OntoNotes 3.0 corpus.

There are two kinds of pipeline systems: word sense disambiguation (WSD) based on semantic role labeling and semantic role labeling (SRL) based on word sense disambiguation. For the using method of word senses, we first only exploit predicate senses (PS) as mentioned by Surdeanu et al. (2008) and Hajič et al. (2009). Then, in order to examine the contribution of word senses except for predicates, we use argument senses (AS) in isolation. Finally, all word senses (PS + AS) were considered.

We can see that when the predicate senses (PS) are used to label semantic role, the performance of semantic role labeling can be improved from 83.97 to 84.17. The conclusion, that the predicate sense can improve semantic role labeling performance, is similar with CoNLL 2008 (Surdeanu et al., 2008) and 2009 (Hajič et al., 2009) shared tasks. However, the improvement is not significant (χ^2-test[8] with $\rho < 0.1$). Additionally, the semantic role labeling can improve the predicate sense disambiguation significantly from 89.37 to 89.53 (z-test with $\rho < 0.1$). The same conclusion was obtained by Dang and Palmer (2005).

However, when we only use argument senses (AS), both of the word sense disambiguation and semantic role labeling performances are almost unchanged (from 89.37 to 89.41 and from 83.97 to 83.94 respectively). For the semantic role labeling task, the reason is that the original lemma and part-of-speech features have been able to describe the argument related information. This kind of sense features is just reduplicate. On the other hand, the argument senses cannot be determined only by the semantic roles. For example, the semantic role "A1" cannot predict the argument sense of "ball" exactly. The predicates must be considered simultaneously.

Therefore, we use the last strategy (PS + AS), which combines the predicate sense and the argument sense together to predict semantic roles. The results show that the performance can be improved significantly (χ^2-test with $\rho < 0.05$) from 83.97 to 84.24. Accordingly, the experiment proves that automatic all word sense disambiguation can further improve the semantic role labeling performance. Different from Che et al. (2010), where the semantic role labeling can be improved with correct word senses about $F_1 = 1$, our improvement is much lower. The main reason is that the performance of our word sense disambiguation with the most basic features is not high enough. Another limitation of the pipeline strat-

[6]Compared with OntoNotes 2.0, the version 3.0 incorporates more corpus.

[7]http://www.dimensionresearch.com/resources/calculators/ztest.html

[8]http://graphpad.com/quickcalcs/chisquared1.cfm

egy is that it is difficult to predict the combination between predicate and argument senses. This is an obvious shortcoming of the pipeline method.

With Markov logic, we can easily join different tasks with global formulas. As shown in Table 4, we use five joint strategies:

1. PS $\Rightarrow$ SRL: means that we jointly disambiguate predicate senses and label semantic roles. Compared with the pipeline PS system, word sense disambiguation performance is unchanged. However, the semantic role labeling performance is improved from 84.17 to 84.27. Compared with the baseline's 83.97, the improvement is significant (χ^2-test with $\rho < 0.05$).

2. AS $\Rightarrow$ SRL: means that we jointly disambiguate argument senses and label semantic roles. Compared with the pipeline AS system, both of word sense disambiguation and semantic role labeling performances are improved (from 89.41 to 89.49 and from 83.94 to 84.16 respectively). Although, the improvement is not significant, it is observed that the joint model has the capacity to improve the performance, especially for semantic role labeling, if we could have a more accurate word sense disambiguation.

3. PS $\Rightarrow$ AS: means that we jointly disambiguate predicate word senses and argument senses. This kind of joint model does not influence the performance of semantic role labeling. The word sense disambiguation outperforms the baseline system from 89.37 to 89.45. The result verifies our assumption that the predicate and argument senses can help each other.

4. PS + AS $\Rightarrow$ SRL: means that we jointly disambiguate all word senses and label semantic roles. Compared with the pipeline method which uses the PS + AS strategy, the joint method can further improve the semantic role labeling (from 84.24 to 84.34). Additionally, it can obtain the predicate and argument senses together. The all word sense disambiguation performance (89.54) is higher than the baseline (89.37) significantly (z-test with $\rho < 0.1$).

5. Fully: finally, we use all of the three global formulas together, i.e., we jointly disambiguate predicate senses, argument senses, and label semantic roles. It fully joins all of the tasks. Both of all word sense disambiguation and semantic role labeling performances can be further improved. Although the improvements are not significant compared with the best pipeline system, they significantly (z-test with $\rho < 0.1$ and χ^2-test with $\rho < 0.01$ respectively) outperform the baseline system. Additionally, the performance of the fully joint system does not outperform partly joint systems significantly. The reason seems to be that there is some overlap among the contributions of the three joint systems.

6 Conclusion

In this paper, we presented a Markov logic model that jointly models all word sense disambiguation and semantic role labeling. We got the following conclusions:

1. The baseline systems with Markov logic is competitive to the state-of-the-art word sense disambiguation and semantic role labeling systems on OntoNotes 3.0 corpus.

2. The predicate sense disambiguation is beneficial to semantic role labeling. However, the automatic argument sense disambiguation itself is harmful to the task. It must be combined with the predicate sense disambiguation.

3. The semantic role labeling not only can help predicate sense disambiguation, but also argument sense disambiguation (a little). In contrast, because of the limitation of the pipeline model, it is difficult to make semantic role labeling to help predicate and argument sense disambiguation simultaneously.

4. It is easy to implement the joint model of all word sense disambiguation and semantic role labeling with Markov logic. More important, the joint model can further improve the performance of the all word sense disambiguation and semantic role labeling than pipeline systems.

Acknowledgement

This work was supported by National Natural Science Foundation of China (NSFC) via grant 60803093, 60975055, the "863" National High-Tech Research and Development of China via grant 2008AA01Z144, and Natural Scientific Research Innovation Foundation in Harbin Institute of Technology (HIT.NSRIF.2009069).

References

Che, Wanxiang, Zhenghua Li, Yongqiang Li, Yuhang Guo, Bing Qin, and Ting Liu. 2009. Multilingual dependency-based syntactic and semantic parsing. In *Proceedings of CoNLL 2009: Shared Task*, pages 49–54, June.

Che, Wanxiang, Ting Liu, and Yongqiang Li. 2010. Improving semantic role labeling with word sense. In *NAACL 2010*, pages 246–249, June.

Crammer, Koby and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Dahlmeier, Daniel, Hwee Tou Ng, and Tanja Schultz. 2009. Joint learning of preposition senses and semantic roles of prepositional phrases. In *Proceedings of EMNLP 2009*, pages 450–458, August.

Dang, Hoa Trang and Martha Palmer. 2005. The role of semantic roles in disambiguating verb senses. In *Proceedings of ACL 2005*, pages 42–49, Morristown, NJ, USA.

Domingos, Pedro and Daniel Lowd. 2009. *Markov Logic: An Interface Layer for Artificial Intelligence*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Finkel, Jenny Rose and Christopher D. Manning. 2009. Joint parsing and named entity recognition. In *Proceedings of NAACL 2009*, pages 326–334, June.

Goldberg, Yoav and Reut Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *Proceedings of ACL 2008*, pages 371–379, June.

Hajič, Jan, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antònia Martí, Lluís Màrquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, and Yi Zhang. 2009. The conll-2009 shared task: Syntactic and semantic dependencies in multiple languages. In *Proceedings of CoNLL 2009: Shared Task*, pages 1–18, June.

Hovy, Eduard, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: The 90% solution. In *Proceedings of NAACL 2006*, pages 57–60, June.

Jiang, Wenbin, Liang Huang, Qun Liu, and Yajuan Lü. 2008. A cascaded linear model for joint chinese word segmentation and part-of-speech tagging. In *Proceedings of ACL 2008*, pages 897–904, June.

Kruengkrai, Canasai, Kiyotaka Uchimoto, Jun'ichi Kazama, Yiou Wang, Kentaro Torisawa, and Hitoshi Isahara. 2009. An error-driven word-character hybrid model for joint chinese word segmentation and pos tagging. In *Proceedings of ACL-IJCNLP 2009*, pages 513–521, August.

Meza-Ruiz, Ivan and Sebastian Riedel. 2009. Jointly identifying predicates, arguments and senses using markov logic. In *Proceedings of NAACL 2009*, pages 155–163, June.

Poon, Hoifung and Pedro Domingos. 2008. Joint unsupervised coreference resolution with Markov Logic. In *Proceedings of EMNLP 2008*, pages 650–659, October.

Punyakanok, Vasin, Dan Roth, and Wen tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2).

Riedel, Sebastian. 2008. Improving the accuracy and efficiency of map inference for markov logic. In *Proceedings of UAI 2008*, pages 468–475. AUAI Press.

Surdeanu, Mihai, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. The conll 2008 shared task on joint parsing of syntactic and semantic dependencies. In *Proceedings of CoNLL 2008*, pages 159–177, August.

Titov, Ivan and Ryan McDonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of ACL 2008*, pages 308–316, June.

Toutanova, Kristina and Colin Cherry. 2009. A global model for joint lemmatization and part-of-speech prediction. In *Proceedings of ACL-IJCNLP 2009*, pages 486–494, August.

Toutanova, Kristina, Aria Haghighi, and Christopher D. Manning. 2008. A global joint model for semantic role labeling. *Computational Linguistics*, 34(2).

Yoshikawa, Katsumasa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with markov logic. In *Proceedings of ACL-IJCNLP 2009*, pages 405–413, August.

Zhang, Yue and Stephen Clark. 2008. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL 2008*, pages 888–896, June.

Zhong, Zhi, Hwee Tou Ng, and Yee Seng Chan. 2008. Word sense disambiguation using OntoNotes: An empirical study. In *Proceedings of EMNLP 2008*, pages 1002–1010, October.

Bipolar Person Name Identification of Topic Documents Using Principal Component Analysis

Chein Chin Chen
Department of Information Management
National Taiwan University
paton@im.ntu.edu.tw

Chen-Yuan Wu
Department of Information Management
National Taiwan University
r97725035@ntu.edu.tw

Abstract

In this paper, we propose an unsupervised approach for identifying bipolar person names in a set of topic documents. We employ principal component analysis (PCA) to discover bipolar word usage patterns of person names in the documents and show that the signs of the entries in the principal eigenvector of PCA partition the person names into bipolar groups spontaneously. Empirical evaluations demonstrate the efficacy of the proposed approach in identifying bipolar person names of topics.

1 Introduction

With the advent of Web2.0, many online collaborative tools, e.g., weblogs and discussion forums are being developed to allow Internet users to express their perspectives on a wide variety of topics via Web documents. One benefit is that the Web has become an invaluable knowledge base for Internet users to learn about a topic comprehensively. Since the essence of Web2.0 is knowledge sharing, collaborative tools are generally designed with few constraints so that users will be motivated to contribute their knowledge. As a result, the number of topic documents on the Internet is growing exponentially. Research subjects, such as topic threading and timeline mining (Nallapati et al., 2004; Feng and Allan, 2007; Chen and Chen, 2008), are thus being studied to help Internet users comprehend numerous topic documents efficiently.

A topic consists of a sequence of related events associated with a specific time, place, and person(s) (Nallapati et al., 2004). Topics that involve bipolar (or competitive) viewpoints are often attention-getting and attract a large number of topic documents. For such topics, identifying the polarity of the named entities, especially person names, in the topic documents would help readers learn the topic efficiently. For instance, for the 2008 American presidential election, Internet users can find numerous Web documents about the Democrat and Republican parties. Identifying important people in the competing parties would help readers form a balanced view of the campaign.

Existing works on topic content mining focus on extracting important themes in topics. In this paper, we propose an unsupervised approach that identifies bipolar person names in a set of topic documents automatically. We employ principal component analysis (PCA) (Smith, 2002) to discover bipolar word usage patterns of important person names in a set of topic documents, and show that the signs of the entries in the principal eigenvector of PCA partition the person names in bipolar groups spontaneously. In addition, we present two techniques, called off-topic block elimination and weighted correlation coefficient, to reduce the effect of data sparseness on person name bipolarization. The results of experiments based on two topic document sets written in English and Chinese respectively demonstrate that the proposed PCA-based approach is effective in identifying bipolar person names. Furthermore, the approach is language independent.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 170–178,
Beijing, August 2010

2 Related Work

Our research is closely related to opinion mining, which involves identifying the polarity (or sentiment) of a word in order to extract positive or negative sentences from review documents (Ganapathibhotla and Liu, 2008). Hatzivassiloglou and McKeown (1997) validated that language conjunctions, such as *and*, *or*, and *but*, are effective indicators for judging the polarity of conjoined adjectives. The authors observed that most conjoined adjectives (77.84%) have the same orientation, while conjunctions that use *but* generally connect adjectives of different orientations. They proposed a log-linear regression model that learns the distributions of conjunction indicators from a training corpus to predict the polarity of conjoined adjectives. Turney and Littman (2003) manually selected seven positive and seven negative words as a polarity lexicon and proposed using pointwise mutual information (PMI) to calculate the polarity of a word. A word has a positive orientation if it tends to co-occur with positive words; otherwise, it has a negative orientation. More recently, Esuli and Sebastiani (2006) developed a lexical resource, called SentiWordNet, which calculates the degrees of objective, positive, and negative sentiments of a synset in WordNet. The authors employed a bootstrap strategy to collect training datasets for the sentiments and trained eight sentiment classifiers to assign sentiment scores to a synset. Kanayama and Nasukawa (2006) posited that polar clauses with the same polarity tend to appear successively in contexts. The authors derived the coherent precision and coherent density of a word in a training corpus to predict the word's polarity. Ganapathibhotla and Liu (2008) investigated comparative sentences in product reviews. To identify the polarity of a comparative word (e.g., longer) with a product feature (e.g., battery life), the authors collected phrases that describe the Pros and Cons of products from Epinions.com and proposed one-side association (OSA), which is a variant of PMI. OSA assigns a positive (negative) orientation to the comparative-feature combination if the synonyms of the comparative word and feature tend to co-occur in the Pros (resp. Cons) phrases.

Our research differs from existing approaches in three respects. First, most works identify the polarity of adjectives and adverbs because the syntactic constructs generally express sentimental semantics. In contrast, our method identifies the polarity of person names. Second, to the best of our knowledge, all existing polarity identification methods require external information sources (e.g., WordNet, manually selected polarity words, or training corpora). However, our method identifies bipolar person names by simply analyzing person name usage patterns in topic documents without using external information. Finally, our method does not require any language constructs, such as conjunctions; hence, it can be applied to different languages.

3 Method

3.1 Data Preprocessing

Given a set of topic documents, we first decompose the documents into a set of non-overlapping *blocks* $B = \{b_1, b_2, \ldots, b_n\}$. A block can be a paragraph or a document, depending on the granularity of PCA sampling. Let $U = \{u_1, u_2, \ldots, u_m\}$ be a set of *textual units* in B. In this study, a unit refers to a person name. Then, the document set can be represented as an *m*x*n unit-block association matrix A*. A column in A, denoted as $\underline{b}_i$, represents a decomposed block i. It is an m-dimensional vector whose j'th entry, denoted as $b_{i,j}$, is the frequency of u_j in b_i. In addition, a row in A, denoted as $\underline{u}_i$, represents a textual unit i; and it is an n-dimensional vector whose j'th entry, denoted as $u_{i,j}$, is the frequency of u_i in b_j.

3.2 PCA-based Person Name Bipolarization

Principal component analysis is a well-known statistical method that is used primarily to identify the most important feature pattern in a high-dimensional dataset (Smith, 2002). In our research, it identifies the most important unit pattern in the topic blocks by first constructing an *m*x*m unit relation matrix R*, in which the (i,j)-entry (denoted as $r_{i,j}$) denotes the correlation coefficient of u_i and u_j. The correlation is computed as follows:

$$r_{i,j} = corr(u_i, u_j) = \frac{\sum_{k=1}^{n}(u_{i,k} - u_i^{\sim}) * (u_{j,k} - u_j^{\sim})}{\sqrt{\sum_{k=1}^{n}(u_{i,k} - u_i^{\sim})^2} * \sqrt{\sum_{k=1}^{n}(u_{j,k} - u_j^{\sim})^2}},$$

where $u_i^{\sim} = 1/n\sum_{k=1}^{n} u_{i,k}$ and $u_j^{\sim} = 1/n\sum_{k=1}^{n} u_{j,k}$ are the average frequencies of units i and j respectively.

The range of $r_{i,j}$ is within [-1,1] and the value represents the degree of correlation between u_i and u_j under the decomposed blocks. If $r_{i,j} = 0$, we say that u_i and u_j are uncorrelated; that is, occurrences of unit u_i and unit u_j in the blocks are independent of each other. If $r_{i,j} > 0$, we say that units u_i and u_j are positively correlated. That is, u_i and u_j tend to co-occur in the blocks; otherwise, both tend to be jointly-absent. If $r_{i,j} < 0$, we say that u_i and u_j are negatively correlated; that is, if one unit appears, the other tends not to appear in the same block simultaneously. Note that if $r_{i,j} \neq 0$, $|r_{i,j}|$ scales the strength of a positive or negative correlation. Moreover, since the correlation coefficient is commutative, $r_{i,j}$ will be identical to $r_{j,i}$ such that matrix R will be symmetric.

A unit pattern is represented as a vector $\underline{v}$ of dimension m in which the i'th entry v_i indicates the weight of i'th unit in the pattern. Since matrix R depicts the correlation of the units in the topic blocks, given a constituent of $\underline{v}$, $\underline{v}^T R \underline{v}$ computes the variance of the pattern to characterize the decomposed blocks. A pattern is important if it characterizes the variance of the blocks specifically. PCA can then identify the most important unit pattern by using the following object function:

$$\max \underline{v}^T R \underline{v},$$
$$\text{s.t. } \underline{v}^T \underline{v} = 1.$$

Without specifying any constraint on $\underline{v}$, the objective function becomes arbitrarily large with large entry values of $\underline{v}$. Constraint $\underline{v}^T \underline{v} = 1$ limits the search space within the set of length-normalized vectors. Chen and Chen (2008) show that the desired $\underline{v}$ for the above constrained optimization problem is the eigenvector of R with the largest eigenvalue. Furthermore, as R is a symmetric matrix, such an eigenvector always exists (Spence et al., 2000) and the optimization problem is solvable.

PCA is not the only method that identifies important textual patterns in terms of eigenvectors. For instance, Gong and Liu (2001), Chen and Chen (2008) utilize the eigenvectors of symmetric matrices to extract salient concepts and salient themes from documents respectively[1]. The difference between PCA and other eigenvector-based approaches lies in the way the unit relation matrix is constructed. PCA calculates $r_{i,j}$ by using the correlation coefficient, whereas the other approaches employ the inner product or cosine formula[2] (Manning et al., 2008) to derive the relationship between textual units. Specifically, the correlation coefficient is identical to the cosine formula if we normalize each unit with its mean:

$$corr(u_i, u_j) = \frac{\sum_{k=1}^{n} (u_{i,k} - u_i^{\sim}) * (u_{j,k} - u_j^{\sim})}{\sqrt{\sum_{k=1}^{n} (u_{i,k} - u_i^{\sim})^2} * \sqrt{\sum_{k=1}^{n} (u_{j,k} - u_j^{\sim})^2}}$$
$$= \frac{\sum_{k=1}^{n} u_{i,k}^{*} * u_{j,k}^{*}}{\sqrt{\sum_{k=1}^{n} u_{i,k}^{*\,2}} * \sqrt{\sum_{k=1}^{n} u_{j,k}^{*\,2}}}$$
$$= \text{cosine}(\underline{u}_i^{*}, \underline{u}_j^{*}),$$

where $\underline{u}_i^{*} = \underline{u}_i - u_i^{\sim}[1,1,\ldots,1]^T$; $\underline{u}_j^{*} = \underline{u}_j - u_j^{\sim}[1, 1,\ldots,1]^T$; and are the mean-normalized vectors of $\underline{u}_i$ and $\underline{u}_j$, respectively. Conceptually, the mean normalization process is the only difference between PCA and other eigenvector-based approaches.

Since the eigenvectors of a symmetric matrix form an orthonormal basis of R^m, they may contain negative entries (Spence et al., 2000). Even though Kleinberg (1999) and Chen and Chen (2008) have shown experimentally that negative entries in an eigenvector are as important as positive entries for describing a certain unit pattern, the meaning of negative entries in their approaches is unexplainable. This is because textual units (e.g., terms, sentences, and documents) in information retrieval are usually characterized by frequency-based metrics, e.g., term frequency, document frequency, or TFIDF (Manning et al., 2008), which can never be negative. In PCA, however, the mean normalization process of the correlation coefficient gives bipolar meaning to positive and negative entries and that helps us partition textual units into bipolar groups in accordance with their signs in $\underline{v}$.

1 The right singular vectors of a matrix A used by Gong and Liu (2001) are equivalent to the eigenvectors of a symmetric matrix $A^T A$ whose entries are the inner products of the corresponding columns of A.

2 The inner product is equivalent to the cosine formula when the calculated vectors are length normalized (Manning et al., 2008).

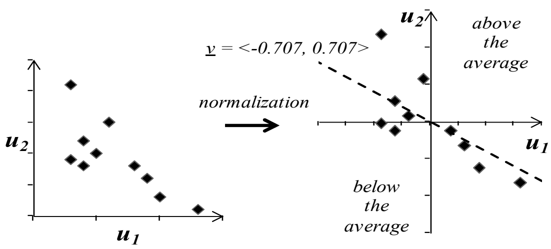


Figure 1. The effect of the mean normalization process.

The synthesized example in Figure 1 illustrates the effect of the normalization process. In this example, we are only interested in textual units u_1 and u_2; the corpus consists of ten blocks. Graphically, each block can be represented as a point in a 2-dimensional vector space. The mean normalization process moves the origin of the 2-dimensional vector space to the centroid of the blocks that makes negative unit values explainable. A negative unit of a block in this normalized vector space indicates that the number of occurrences of the unit in the block is less than the unit's average; by contrast, a positive unit means that the number of occurrences of the unit in a block is above the average. In the figure, the most important unit pattern $\underline{v}$ <-0.707, 0.707> calculated by PCA is represented by the dashed line. The signs of $\underline{v}$'s entries indicate that the occurrence of u_1 will be lower than the average if u_2 occurs frequently in a block. In addition, as the signs of entries in an eigenvector are invertible (Spence et al., 2000), the constituent of $\underline{v}$ also claims that if u_1 occurs frequently in a block, then the probability that we will observe u_2 in the same block will be lower than expected. The instances of bipolar word usage behavior presented in $\underline{v}$ are consistent with the distribution of the ten blocks. As mentioned in Section 2, Kanayama and Nasukawa (2006) validated that polar text units with the same polarity tend to appear together to make contexts coherent. Consequently, we believe that the signs in PCA's principal eigenvector are effective in partitioning textual units into bipolar groups.

3.3 Sparseness of Textual Units

A major problem with employing PCA to process textual data is the sparseness of textual units. To illustrate this problem, we collected 411 news documents about the 2009 NBA Finals from Google News and counted the frequency that each person name occurred in the documents. We also evaluate the documents in the experiment section to determine if the proposed approach is capable of bipolarizing the person names into the teams that played in the finals correctly. We rank the units according to their frequencies and list the frequencies in descending order in Figure 2. The figure shows that the frequency distribution follows Zipf's law (Manning et al., 2008); and for most units, the distribution in a block will be very sparse.

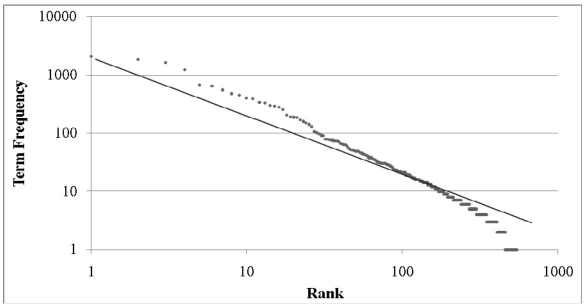


Figure 2. The rank-frequency distribution of person names on logarithmic scales (base 10).

We observe that a unit will not to occur in a block in the following three scenarios. 1) The polarity of the block is the opposite of the polarity of the unit. For instance, if the unit represents a player in one team and the block narrates information about the other team, the block's author would not mention the unit in the block to ensure that the block's content is coherent. 2) Even if the polarity of a block is identical to that of the unit; the length of the block may not be sufficient to contain the unit. 3) The block is off-topic so the unit will not appear in the block. In the last two scenarios, the absence of units will impact the estimation of the correlation coefficient. To alleviate the problem, we propose two techniques, the weighted correlation coefficient and off-block elimination, which we describe in the following sub-sections.

Weighted Correlation Coefficient

The so-called data sparseness problem in scenario 2 affects many statistical information retrieval and language models (Manning et al., 2008). For units with the same polarity, data sparseness could lead to underestimation of their correlations because the probability that the units will occur together is reduced. Conversely, for uncorrelated units or units with opposite polarities, data sparseness may lead to overestimation of their correlations because they are frequently jointly-absent in the decomposed blocks. While smoothing approaches, such as Laplace's law (also known as adding-one smoothing), have been developed to alleviate data sparseness in language models (Manning et al., 2008), they are not appropriate for PCA. This is because the correlation coefficient of PCA measures the divergence between units from their means, so adding one to each block unit will not change the divergence. To summarize, data sparseness could influence the correlation coefficient when units do not co-occur. Thus, for two units u_i and u_j, we separate B into co-occurring and non-co-occurring parts and apply the following weighted correlation coefficient:

$$corr_w(u_i,u_j)= \left(\begin{array}{l} (1-\alpha)\sum_{b\in co(i,j)}(u_{i,b}-\tilde{u_i})*(u_{j,b}-\tilde{u_j})+ \\ \qquad \alpha\sum_{b\in B-co(i,j)}(u_{i,b}-\tilde{u_i})*(u_{j,b}-\tilde{u_j}) \end{array} \right) \Bigg/ \sqrt{(1-\alpha)\sum_{b\in co(i,j)}(u_{i,b}-\tilde{u_i})^2+\alpha\sum_{b\in B-co(i,j)}(u_{i,b}-\tilde{u_i})^2} * \sqrt{(1-\alpha)\sum_{b\in co(i,j)}(u_{j,b}-\tilde{u_j})^2+\alpha\sum_{b\in B-co(i,j)}(u_{j,b}-\tilde{u_j})^2},$$

where $corr_w(u_i,u_j)$ represents the weighted correlation coefficient between units i and j; and $co(i,j)$ denotes the set of blocks in which units i and j co-occur. The range of parameter α is within [0,1]. It weights the influence of non-co-occurring blocks when calculating the correlation coefficient. When $\alpha = 0.5$, the equation is equivalent to the standard correlation coefficient; and when $\alpha = 0$, the equation only considers the blocks in which units i and j co-occur. Conversely, when $\alpha = 1$, only non-co-occurring blocks are employed to calculate the units' correlation. In the experiment section, we will examine the effect of α on bipolar person name identification.

Off-topic Block Elimination

Including off-topic blocks in PCA will lead to overestimation of the correlation between units. This is because units are usually jointly-absent from off-topic blocks that make uncorrelated or even negatively correlated units positively correlated. To eliminate the effect of off-topic blocks on unit bipolarization, we construct a centroid of all the decomposed blocks by averaging $\underline{b}_i$'s. Then, blocks whose cosine similarity to the centroid is lower than a predefined threshold β are excluded from calculation of the correlation coefficient.

4 Performance Evaluations

In this section, we evaluate two topics with bipolar (or competitive) viewpoints to demonstrate the efficacy of the proposed approach.

4.1 The 2009 NBA Finals

For this experiment, we collected 411 news documents about the 2009 NBA Finals from Google News during the period of the finals (from 2009/06/04 to 2009/06/16). The matchup of the finals was Lakers versus Orlando Magic. In this experiment, a block is a topic document, as paragraph tags are not provided in the evaluated documents. First, we parsed the blocks by using Stanford Named Entity Recognizer[3] to extract all possible named entities. We observed that the parser sometimes extracted false entities (such as Lakers Kobe) because the words in the headlines were capitalized and that confused the parser. To reduce the effect of false extraction by the parser, we examined the extracted named entities manually. After eliminating false entities, the dataset comprised 546 unique named entities; 538 were person names and others represented organizations, such as basketball teams and basketball courts. To examine the effect of the weighted correlation coefficient, parameter α is set between 0 and 1, and increased in increments of 0.1; and the threshold β used by off-topic block elimination is set at 0.3. The frequency distribution of the person names, shown in Figure 2, indicates that many of the person names rarely appeared in the examined blocks, so their distribution was too sparse for PCA. Hence, in the following subsections, we sum the frequencies of the 538 person names in the examined blocks. We select the first k frequent person names, whose accumulated term frequencies reach 60% of the total frequencies, for evaluation. In other words, the evaluated person names account for 60% of the person name occurrences in the examined blocks.

For each parameter setting, we perform principal component analysis on the examined blocks and the selected entities, and partition the entities into two bipolar groups according to

3 http://nlp.stanford.edu/software/CRF-NER.shtml

their signs in the principal eigenvector. To evaluate the accuracy rate of bipolarization, we need to label the team of each bipolar group. Then, the accuracy rate is the proportion of the entities in the groups that actually belong to the labeled teams. Team labeling is performed by examining the person names in the larger bipolarization group. If the majority of the entities in the group belong to the Lakers (Magic), we label the group as Lakers (Magic) and the other group as Magic (Lakers). If the two bipolar groups are the same size, the group that contains the most Lakers (Magic) entities is labeled as Lakers (Magic), and the other group is labeled as Magic (Lakers). If both groups contain the same number of Lakers (Magic) entities, we randomly assign team labels because all assignments produce the same accuracy score. To the best of our knowledge, there is no similar work on person name bipolarization; therefore, for comparison, we use a baseline method that assigns the same polarity to all the person names.

Magic		Lakers	
Dwight Howard	0.0884	Derek Fisher	-0.0105
Hedo Turkoglu	0.1827	Kobe Bryant	-0.2033
Jameer Nelson	0.3317	Lamar Odom	-0.1372
Jeff Van Gundy*+	0.3749	LeBron James*^	-0.0373
Magic Johnson*	0.3815	Mark Jackson*^	-0.2336
Rafer Alston	0.3496	Pau Gasol	-0.1858
Rashard Lewis	0.1861	Paul Gasol*+	-0.1645
Stan Van Gundy	0.4035	Phil Jackson	-0.2553

Table 1. The bipolarization results for NBA person names. ($\alpha = 0.8$ and $\beta = 0.3$)

Table 1 shows the bipolarization results for frequent person names in the dataset. The parameter α is set at 0.8 because of its superior performance. The left-hand column of the table lists the person names labeled as Magic and their entry values in the principal eigenvector; and the right-hand column lists the person names labeled as Lakers. It is interesting to note that the evaluated entities contain person names irrelevant to the players in the NBA finals. For instance, the frequency of Magic Johnson, an ex-Lakers player, is high because he constantly spoke in support of the Lakers during the finals. In addition, many documents misspell Pau Gasol as Paul Gasol. Even though the names refer to the same player, the named entity recognizer parses them as distinct entities. We propose two evaluation strategies, called *strict evaluation* and *non-strict evaluation*. The strict evaluation strategy treats the person names that do not refer to the players, coaches in the finals as false positives. Under the non-strict strategy, the person names that are closely related to Lakers or Magic players, such as a player's relatives or misspellings, are deemed true positives if they are bipolarized into the correct teams. In Table 1, a person name annotated with the symbol * indicates that the entity is bipolarized incorrectly. For instance, Magic Johnson is not a member of Magic. The symbol ^ indicates that the person name is neutral (or irrelevant) to the teams in the finals. In addition, the symbol + indicates that the person name represents a relative of a member of the team he/she is bipolarized to; or the name is a misspelling, but it refers to a member of the bipolarized team. This kind of bipolarization is correct under the non-strict evaluation strategy. As shown in Table 1, the proposed method bipolarizes the important persons in the finals correctly without using any external information source. The accuracy rates of strict and non-strict evaluation are 68.8% and 81.3% respectively. The rates are far better than those of the baseline method, which are 37.5% and 43.8% respectively. If we ignore the neutral entities, which are always wrong no matter what bipolarization approach is employed, the strict and non-strict accuracies are 78.6% and 92.9% respectively. In the non-strict evaluation, we only mis-bipolarized Magic Johnson as Magic. The mistake also reflects a problem with person name resolution when the person names that appear in a document are ambiguous. In our dataset, the word 'Magic' sometimes refers to Magic Johnson and sometimes to Orlando Magic. Here, we do not consider a sophisticated person name resolution scheme; instead, we simply assign the frequency of a person name to all its specific entities (e.g., Magic to Magic Johnson, and Kobe to Kobe Bryant) so that specific person names are frequent enough for PCA. As a result, Magic Johnson tends to co-occur with the members of Magic and is incorrectly bipolarized to the Magic team. Another interesting phenomenon is that LeBron James (a player with Cavaliers) is incorrectly bipolarized to Lakers. This is because Kobe Bryant (a player with Lakers) and LeBron James were rivals for the most valuable player (MVP) award in the 2009 NBA season. The documents that mentioned Kobe Bryant during the finals often compared him with LeBron

James to attract the attention of readers. As the names often co-occur in the documents, LeBron James was wrongly classified as a member of Lakers.

Figures 3 and 4 illustrate the effects of the weighted correlation coefficient and off-topic block elimination on NBA person name bipolarization. As shown in the figures, eliminating off-topic blocks generally improves the system performance. It is noteworthy that, when off-topic blocks are eliminated, large α values produce good bipolarization performances. As mentioned in Section 3.3, a large α implies that non-co-occurring blocks are important for calculating the correlation between a pair of person names. When off-topic blocks are eliminated, the set of non-co-occurring blocks specifically reveals opposing or jointly-absent relationships between entities. Therefore, the bipolarization performance improves as α increases. Conversely, when off-topic blocks are not eliminated, the set of non-co-occurring blocks will contain off-topic blocks. As both entities in a pair tend to be absent in off-topic blocks, a large α value will lead to overestimation of the correlation between bipolar entities. Consequently, the bipolarization accuracy decreases as α increases. It is also interesting to note that the bipolarization performance decreases as α decreases. We observed that some of the topic documents are recaps of the finals, which tend to mention Magic and Lakers players together. As a small α value makes co-occurrence blocks important, recap-style documents will overestimate the correlation between bipolar entities. Consequently, the bipolarization performance is inferior when α is small.

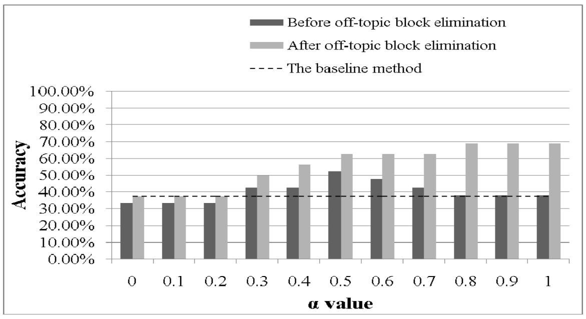


Figure 3. The effects of the weighted correlation coefficient and off-topic block elimination on NBA person name bipolarization. (Strict)

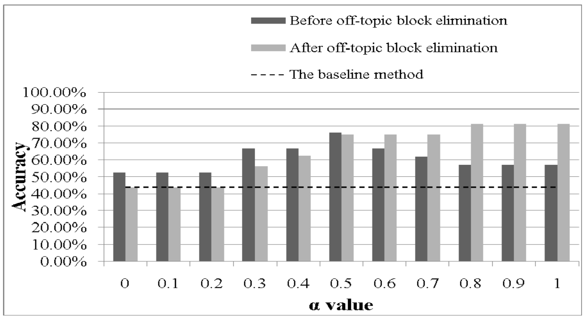


Figure 4. The effects of the weighted correlation coefficient and off-topic block elimination on NBA person name bipolarization. (Non-strict)

4.2 Taiwan's 2009 Legislative By-Elections

For this experiment, we evaluated Chinese news documents about Taiwan's 2009 legislative by-elections, in which two major parties, the Democratic Progressive Party (DPP) and the KouMinTang (KMT), campaigned for three legislative positions. Since the by-elections were regional, not many news documents were published during the campaign. In total, we collected 89 news documents that were published in The Liberty Times [4] during the election period (from 2009/12/27 to 2010/01/11). Then, we used a Chinese word processing system, called Chinese Knowledge and Information Processing (CKIP)[5], to extract possible Chinese person names in the documents. Once again, the names were examined manually to remove false extractions. The dataset comprised 175 unique person names. As many of the names only appeared once, we selected the first k frequent person names whose accumulated frequency was at least 60% of the total term frequency count of the person names for evaluation. We calculated the accuracy of person name bipolarization by the same method as the NBA experiment in order to assess how well the bipolarized groups represented the KMT and the DPP. As none of the selected names were misspelled, we do not show the non-strict accuracy of bipolarization. The threshold β is set at 0.3, and each block is a topic document.

Table 2 shows the bipolarization results for the frequent person names of the candidates of the respective parties, the party chair persons, and important party staff members. The accuracy rates of the bipolarization and the baseline me-

[4] http://www.libertytimes.com.tw/index.htm
[5] http://ckipsvr.iis.sinica.edu.tw/

thods are 70% and 50%, respectively. It is noteworthy that the chairs of the DPP and the KMT, who are Ing-wen Tsai and Ying-jeou Ma respectively, are correctly bipolarized. We observed that, during the campaign, the chairs repeatedly helped their respective party's candidates gain support from the public. As the names of the chairs and the candidates often co-occur in the documents, they can be bipolarized accurately. We also found that our approach bipolarized two candidates incorrectly if the competition between them was fierce. For instance, Kun-cheng Lai and Li-chen Kuang campaigned intensively for a single legislative position. As they often commented on each other during the campaign, they tend to co-occur in the topic documents. PCA therefore misclassifies them as positively correlated and incorrectly groups Kun-cheng Lai with the KMT party.

KMT (國民黨)		DPP (民進黨)	
Kun-cheng Lai (賴坤成)*	0.39	Wen-chin Yu (余文欽)*	-0.56
Li-chen Kuang (鄺麗貞)	0.40	Den-yih Wu (吳敦義)*	-0.03
Li-ling Chen (陳麗玲)	0.01	Chao-tung Chien (簡肇棟)	-0.56
Ying-jeou Ma (馬英九)	0.05	Ing-wen Tsai (蔡英文)	-0.17
		Tseng-chang Su (蘇貞昌)	-0.01
		Jung-chung Kuo (郭榮宗)	-0.01

Table 2. The bipolarization results for the election dataset. ($\alpha = 0.7$)

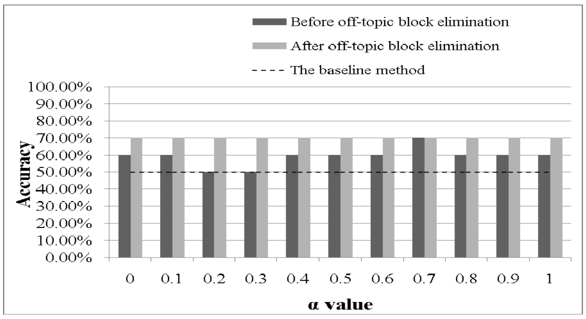


Figure 5. The effects of the weighted correlation coefficient and off-topic block elimination.

Figure 5 shows that off-topic block elimination is effective in person name bipolarization. However, the weighted correlation coefficient only improves the bipolarization performance slightly. We have investigated this problem and believe that the evaluated person names in the documents are frequent enough to prevent the data sparseness problem. While the weighted correlation coefficient does not improve the bipolarization performance significantly, the proposed PCA-based approach can still identify the bipolar parties of important persons accurately. Unlike the results in the last section, the accuracy rate in this experiment does not decrease as α decreases. This is because the topic documents generally report news about a single party. As the documents rarely recap the activities of parties, the co-occurrence blocks accurately reflect the bipolar relationship between the persons. Hence, a small α value can identify bipolar person names effectively.

The evaluations of the NBA and the election datasets demonstrate that the proposed PCA-based approach identifies bipolar person names in topic documents effectively. As the writing styles of topic documents in different domains vary, the weighted correlation coefficient may not always improve bipolarization performance. However, because we eliminate off-topic blocks, a large α value always produces superior bipolarization performances.

5 Conclusion

In this paper, we have proposed an unsupervised approach for identifying bipolar person names in topic documents. We show that the signs of the entries in the principal eigenvector of PCA can partition person names into bipolar groups spontaneously. In addition, we introduce two techniques, namely the weighted correlation coefficient and off-topic block elimination, to address the data sparseness problem. The experiment results demonstrate that the proposed approach identifies bipolar person names of topics successfully without using any external knowledge; moreover, it is language independent. The results also show that off-topic block elimination along with a large α value for the weighted correlation coefficient generally produce accurate person name bipolarization. In the future, we will integrate text summarization techniques with the proposed bipolarization method to provide users with polarity-based topic summaries. We believe that summarizing important information about different polarities would help users gain a comprehensive knowledge of a topic.

Acknowledge

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported in part by NSC 97-2221-E-002-225-MY2.

References

Chen, Chien Chin and Meng Chang Chen. 2008. TSCAN: a novel method for topic summarization and content anatomy. In *Proceedings of the 31st annual international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 579-586.

Esuli, Andrea and Fabrizio Sebastiani. 2006. SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation*.

Feng, Ao and James Allan. 2007. Finding and Linking Incidents in News. In *Proceedings of the sixteenth ACM Conference on information and knowledge management*, pages 821-830.

Ganapathibhotla, Murthy and Bing Liu. 2008. Mining Opinions in Comparative Sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 241-248.

Gong, Yihong and Xin Liu. 2001. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 19-25.

Hatzivassiloglou, Vasileios and Kathleen R. McKeown. 1997. Predicting the Semantic Orientation of Adjectives. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 174-181.

Kanayama, Hiroshi and Tetsuya Nasukawa. 2006. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 355-363.

Kleinberg, Jon M.. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM 46, 5*, pages 604-632.

Manning, Christopher D., Prabhakar Raghavan and Hinrich Schutze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Nallapati, Ramesh, Ao Feng, Fuchun Peng and James Allan. 2004. Event Threading within News Topics. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 446-453.

Smith, Lindsay I.. 2002. *A Tutorial on Principal Components Analysis*. Cornell University.

Spence, Lawrence E., Arnold J. Insel and Stephen H. Friedberg. 2000. *Elementary Linear Algebra, A Matrix Approach*. Prentice Hall.

Turney, Peter D., and Michael L. Littman. 2003. Measuring Praise and Criticism: Inference of Semantic Orientation from Association. *ACM Transactions on Information Systems (TOIS)*, pages 315-346.

Emotion Cause Detection with Linguistic Constructions

Ying Chen[*†], Sophia Yat Mei Lee[†], Shoushan Li[†], Chu-Ren Huang[†]
[*] Dep. of Computer Engineering
China Agricultural University
[†]Dep. of Chinese and Bilingual Studies
The Hong Kong Polytechnic University
{chenying3176,sophiaym,shoushan.li,churenhuang}@gmail.com

Abstract

This paper proposes a multi-label approach to detect emotion causes. The multi-label model not only detects multi-clause causes, but also captures the long-distance information to facilitate emotion cause detection. In addition, based on the linguistic analysis, we create two sets of linguistic patterns during feature extraction. Both manually generalized patterns and automatically generalized patterns are designed to extract general cause expressions or specific constructions for emotion causes. Experiments show that our system achieves a performance much higher than a baseline model.

1 Introduction

Text-based emotion processing has been a center of attention in the NLP field in the past few years. Most previous researches have focused on detecting the surface information of emotions, especially emotion classes, e.g., "happiness" and "anger" (Mihalcea and Liu 2006, Strapparava and Mihalcea 2008, Abbasi et al, 2008, Tokuhisa et al. 2008). Although most emotion theories recognize the important role of causes in emotion analysis (Descartes, 1649; James, 1884; Plutchik 1980, Wierzbicka 1999), very few studies explore the interactions between emotion and causes. Emotion-cause interaction is the eventive relation which potentially yields the most crucial information in terms of information extraction. For instance, knowing the existence of an emotion is often insufficient to predict future events or decide on the best reaction. However, if the emotion cause is known in addition to the type of emotion, prediction of future events or assessment of potential implications can be done more reliably. In other words, when emotion is treated as an event, causal relation is the pivotal relation to discover. In this paper, we explore one of the crucial deep level types of information of emotion, i.e. cause events.

Our study focuses on explicit emotions in which emotions are often presented by emotion keywords such as "*shocked*" in "*He was shocked after hearing the news*". Emotion causes are the explicitly expressed propositions that evoke the presence of the corresponding emotions. They can be expressed by verbs, nominalizations, and nominals. Lee et al. (2010a) explore the causes of explicit emotions by constructing a Chinese emotion cause corpus. Based on this corpus, we formalize the emotion cause detection problem through extensive data analysis. We find that ~14% emotion causes are complicated events containing multi-clauses, to which previous cause detection systems can hardly be applied directly. Most previous cause detection systems focus on the causal relation between a pair of small-size text units, such as clauses or phrases. They are thus not able to detect emotion causes that are multi-clauses. In this paper, we formalize emotion cause detection as a multi-label classification task (i.e. each instance may contain more than one label), which allows us to capture long-distance information for emotion cause detection.

In term of feature extraction, as emotion cause detection is a case of cause detection, some typical patterns used in existing cause detection systems, e.g., "*because*" and "*thus*", can be adopted. In addition, various linguistic cues are examined which potentially indicate emotion causes, such as causative verbs and epistemic markers (Lee at al. 2010a). Then some linguistic patterns of emotion causes are manu-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 179–187,
Beijing, August 2010

ally generalized by examining the linguistic context of the empirical data (Lee et al., 2010b). It is expected that these manually generalized patterns often yield a low-coverage problem. Thus, we extracted features which enable us to automatically capture more emotion-specific constructions. Experiments show that such an integrated system with various linguistic features performs promisingly well. We believe that the present study should provide the foundation for future research on emotion analysis, such as the detection of implicit emotion or cause.

The paper is organized as follows. Section 2 discusses the related work on cause-effect detection. Section 3 briefly describes the emotion cause corpus, and then presents our data analysis. Section 4 introduces the multi-label classification system for emotion cause detection. Section 5 describes the two kinds of features for our system, one is based on hand-coded patterns and the other is the generalized features. Section 6 presents the evaluation and performance of our system. Section 7 highlights our main contributions and the possible future work.

2 Related Work

Most previous studies on textual emotion processing focus on emotion recognition or classification given a known emotion context (Mihalcea and Liu 2006, Strapparava and Mihalcea 2008, Abbasi et al, 2008, Tokuhisa et al. 2008). However, the performance is far from satisfactory. One crucial problem in these works is that they limit the emotion analysis to a simple classification and do not explore the underlying information regarding emotions. Most theories conclude that emotions are often invoked by the perception of external events. An effective emotion recognition model should thus take this into account.

To the best of our knowledge, little research has been done with respect to emotion cause detection. Lee et al. (2010a) first investigate the interactions between emotions and the corresponding causes from a linguistic perspective. They annotate a small-scale emotion cause corpus, and identify six groups of linguistic cues facilitating emotion cause detection. Based on these findings, they develop a rule-based system for automatic emotion cause detection (Lee et al., 2010b).

Emotion cause detection can be considered as a kind of causal relation detection, which has been intensively studied for years. Most previous cause detection studies focus on a specific domain, such as aviation (Persing and Ng, 2009) and finance (Low, et al., 2001). Few works (Marcu and Echihabi, 2002; Girju, 2003; Chang and Choi, 2005) examine causal relation for open domains.

In recognizing causal relations, most existing systems involve two steps: 1) cause candidate identification; 2) causal relation detection. To simplify the task, most systems omit the step of identifying cause candidates. Instead, they often predefine or filter out possible causes based on domain knowledge, e.g., 14 kinds of cause types are identified for aviation incidents (Persing and Ng, 2009). For events without specific domain information, open-domain systems choose to limit their cause candidate. For example, the cause-effect pairs are limited to two noun phrases (Chang and Choi, 2005; Girju, 2003), or two clauses connected with fixed conjunction words (Marcu and Echihabi, 2002).

Given pairs of cause-effect candidates, causal relation detection is considered as a binary classification problem, i.e. “causal” vs. “non-causal”. In general, there are two kinds of information extracted to identify the causal relation. One is patterns or constructions expressing a cause-effect relation (Chang and Choi, 2005; Girju, 2003), and the other is semantic information underlying in a text (Marcu and Echihabi, 2002; Persing and Ng, 2009), such as word pair probability. Undoubtedly, the two kinds of information usually interact with each other in a real cause detection system.

In the literature, the three common classification methods, i.e. unsupervised, semi-supervised, and supervised, have all been used for cause detection systems. Marcu and Echihabi (2002) first collected a cause corpus using an unsupervised approach with the help of several conjunction words, such as “*because*” and “*thus*”, and determined the causal relation for a clause pair using the word pair probability. Chang and Choi (2005) used a semi-supervised method to recursively learn lexical patterns for cause recognition based on syntactic trees. Bethard and Martin (2008) put various causal information in a supervised classifier, such as the temporal information and syntactic information.

For our emotion cause detection, several practical issues need to be investigated and resolved. First, for the identification of cause candidates, we need to define a reasonable span of a cause. Based on our data analysis, we find that emotion causes often appear across phrases or even clauses. Second, although in emotion cause detection the effect is fixed, the cause is open-domain. We also notice that besides the common patterns, emotion causes have their own expression patterns. An effective emotion cause detection system should take them into account.

3 Corpus Analysis

In this section, we briefly introduce the Chinese emotion cause corpus (Lee et al., 2010a), and discuss emotion cause distribution.

3.1 Emotion Cause corpus

Lee at al. (2010a) made the first attempt to explore the correlation between emotions and causes, and annotate a Chinese emotion cause corpus. The emotion cause corpus focuses on five primary emotions, namely “happiness”, “sadness”, “fear”, “anger”, and “surprise”. The emotions are explicitly expressed by emotion keywords, e.g., *gao1xing4* “happy”, *shang1xin1* “sad”, etc. The corpus is created as follows.

1. 6,058 entries of Chinese sentences are extracted from the Academia Sinica Balanced Corpus of Mandarin Chinese (Sinica Corpus) with the pattern-match method as well as the list of 91 Chinese primary emotion keywords (Chen et al., 2009). Each entry contains the focus sentence with the emotion keyword “<FocusSentence>” plus the sentence before “<PrefixSentence>” and after “<SuffixSentence>” it. For each entry, the emotion keywords are indexed since more than one emotion may be presented in an entry;
2. Some preprocessing, such as balancing the number of entry among emotions, is done to remove some entries. Finally, 5,629 entries remain;
3. Each emotion keyword is annotated with its corresponding causes if existing. An emotion keyword can sometimes be associated with more than one cause, in such a case, both causes are marked. Moreover, the cause type is also identified, which is either a nominal event or a verbal event (a verb or a nominalization).

Lee at al. (2010a) notice that 72% of the extracted entries express emotions, and 80% of the emotional entries have a cause.

3.2 The Analysis of Emotion Causes

To have a deeper understanding of emotion cause detection, we take a closer look at the emotion cause distribution, including the distribution of emotion cause occurrence and the distribution of emotion cause text.

The occurrence of emotion causes: According to most emotion theories, an emotion is generally invoked by an external event. The corpus shows that, however, 20% of the emotional entries have no cause. Entries without causes explicitly expressed are mainly due to the following reasons:

i) There is not enough contextual information, for instance the previous or the suffix sentence is interjections, e.g., *en heng* “aha”;

ii) When the focus sentence is the beginning or the ending of a paragraph, no prefix sentence or suffix sentence can be extracted as the context. In this case, the cause may be beyond the context;

iii) The cause is obscure, which can be very abstract or even unknown reasons.

The emotion cause text: A cause is considered as a proposition. It is generally assumed that a proposition has a verb which optionally takes a noun occurring before it as the subject and a noun after it as the object. However, a cause can also be expressed as a nominal. In other words, both the predicate and the two arguments are optional provided that at least one of them is present. Thus, the fundamental issue in designing a cause detection system is the definition of the span of a cause text. As mentioned, most previous studies on causal relations choose to ignore the identification of cause candidates. In this paper, we first analyze the distribution of cause text and then determine the cause candidates for an emotion.

Based on the emotion cause corpus, we find that emotion causes are more likely to be expressed by verbal events than nominal events (85% vs. 15%). Although a nominalization (a kind of verbal events) is usually a noun phrase, a proposition containing a verb plays a salient role in the expressions of emotion causes, and thus a cause candidate are more likely to be a clause-based unit.

In addition, the actual cause can sometimes be too long and complicated, which involves several events. In order to explore the span of a cause text, we do the following analysis.

Table 1: The clause distribution of cause texts

Position	Cause (%)	Position	Cause (%)
Left_0	12.90	Right _0	15.54
Left_1	31.37	Right _1	9.55
Left_2	13.31	Right_n (n>1)	9.18
Left_n (n>2)	10.15		
Total	67.73		32.27

Table 2: The multi-clause distribution of cause text

Same clause	%	Cross-clauses	%
Left_0	16.80	Left_2_1_0	0.25
Left_1	31.82	Left_2_1	10.84
Left_2	7.33	Left_1_0	0.62
Right _0	18.97	Right_0_1	2.55
Right _1	10.59		
Total	85.75		14.25

Firstly, for each emotion keyword, an entry is segmented into clauses with four punctuations (i.e. commas, periods, question marks and exclamation marks), and thus an entry becomes a list of cause candidates. For example, when an entry has four clauses, its corresponding list of cause candidates contains five text units, i.e. <left_2, left_1, left_0, right_0, right_1>. If we assume the clause where emotion keyword locates is a focus clause, 'left_2' and 'left_1' are previous two clauses, and 'right_1' is the following one. 'left_0' and 'right_0' are the partial texts of the focus clause, which locate in the left side of and the right side of the emotion keyword, respectively. Moreover, a cause candidate must contain either a noun or a verb because a cause is either a verbal event or a nominal event; otherwise, it will be removed from the list.

Secondly, we calculate whether a cause candidate overlaps with the real cause, as shown in Table 1. We find that emotion causes are more likely to occur in the left of emotion keyword. This observation is consistent with the fact that an emotion is often trigged by an external happened event. Thirdly, for all causes occurring between 'left_2' and 'right_1', we calculate whether a cause occurs across clauses, as in Table 2. We observe that most causes locate within the same clause of the representation of the emotion (85.57%). This suggests that a clause may be the most appropriate unit to detect a cause.

4 Emotion Cause Detection Based on Multi-label Classification

A cause detection system is to identify the causal relation between a pair of two text units. For emotion cause detection, one of the two text units is fixed (i.e. the emotion keyword), and therefore the remaining two unresolved issues are the identification of the other text unit and the causal relation.

From the above data analysis, there are two observations. First, most emotion causes are verbal events, which are often expressed by a proposition (or a clause). Thus, we define another text unit as a clause, namely a cause candidate. Second, as most emotion causes occur between 'left_2' and 'right_1' (~80%), we define the cause candidates for an emotion as <left_2, left_1, left_0, right_0, right_1>.

Differing from the existing cause systems, we formalize emotion cause detection as a multi-label problem. In other words, given an emotion keyword and its context, its label is the locations of its causes, such as "left_1, left_0". This multi-label-based formalization of the cause detection task has two advantages. First, it is an integrated system detecting causes for an emotion from the contextual information. In most previous cause detection systems, a causal relation is identified based on the information between two small text units, i.e. a pair of clauses or noun phrases, and therefore it is often the case that long-distance information is missed. Second, the multi-label-based tagging is able to capture the relationship between two cause candidates. For example, "left_2" and "left_1" are often combined as a complicated event as a cause.

As a multi-label classification task, every multi-label classifier is applicable. In this study, we use a simple strategy: we treat each possible combination of labels appearing in the training data as a unique label. Note that an emotion without causes is labeled as "None". This converts multi-label classification to single-label classification, which is suitable for any multi-class classification technologies. In particular, we choose a Max Entropy tool, Mallet[1], to perform the classification.

5 Linguistic Features

As explained, there are basically two kinds of features for cause detection, namely pattern-based features and semantic-based features. In this study, we develop two sets of patterns based on linguistic analysis: one is a set of manually generalized patterns, and the other contains automatically generalized patterns. All of these patterns explore causal constructions either for general causal relations or for specific emotion cause relations.

5.1 Linguistic Cues

Based on the linguistic analysis, Lee et al. (2010a) identify six groups of linguistic cue words that are highly collocated with emotion causes, as shown in Table 3. Each group of the linguistic cues serves as an indicator marking the causes in different emotional constructions. In this paper, these groups of linguistic cues are reinterpreted from the computational perspective, and are used to develop pattern-based features for the emotion cause detection system.

Table 3: Linguistic cue words for emotion cause detection (Lee et al. 2010a)

Group	Cue Words
I: Prepositions	'for' as in 'I will do this for you': *wei4*, *wei4le* 'for' as in 'He is too old for the job': *dui4*, *dui4yu2* 'as': *yi3*
II: Conjunctions	'because': *yin1*, *yin1wei4*, *you2yu2* 'so': *yu1shi4*, *suo3yi3*, *yin1er2* 'but': *ke3shi4*
III: Light Verbs	"to make": *rang4*, *ling4*, *shi3*
IV: Reported Verbs	'to think about': *xiang3dao4*, *xiang3qi3*, *yi1xiang3*, *xiang3 lai2* 'to talk about': *shuo1dao4*, *shuo1qi3*, *yi1shuo1*, *jiang3dao4*, *jiang3qi3*, *yi1jiang3*, *tan2dao4*, *tan2qi3*, *yi1tan2*, *ti2dao4*, *ti2qi3*, *yi1ti2*
V: Epistemic Markers	'to hear': *ting1*, *ting1dao4*, *ting1shuo1* 'to see': *kan4*, *kan4dao4*, *kan4jian4*, *jian4dao4*, *jian4*, *yan3kan4*, *qiao2jian4* 'to know': *zhi1dao4*, *de2zhi1*, *de2xi1*, *huo4zhi1*, *huo4xi1*, *fa1xian4*, *fa1jue2* 'to exist': *you3*
VI: Others	'is': *deshi4* 'say': *deshuo1* 'at': *yu2* 'can': *neng2*

For emotion cause processing, Group I and II contain cues which are for general cause detection, and while Group III, IV and V include cues specifically for emotion cause detection. Group VI includes other linguistic cues that do not fall into any of the five groups.

Group I covers some prepositions which all roughly mean 'for', and Group II contains the conjunctions that explicitly mark the emotion cause. Group I is expected to capture the prepositions constructions in the focus clause where the emotion keyword locates. Group II tends to capture the rhetorical relation expressed by conjunction words so as to infer causal relation among multi-clauses. These two groups are typical features for general cause detection.

Group III includes three common light verbs which correspond to the English equivalents "to make" or "to cause". Although these light verbs themselves do not convey any concrete meaning, they are often associated with several constructions to express emotions and at the same time indicate the position of emotion causes. For example, "*The birthday party made her happy*".

One apparent difference between emotion causes and general causes is that emotions are often triggered by human activities or the perception of such activities, e.g., "*glad to say*" or "*glad to hear*". Those human activities are often strong indicators for the location of emotion

[1] http://mallet.cs.umass.edu/

causes. Group IV and V are used to capture this kind of information. Group IV is a list of verbs of thinking and talking, and Group V includes four types of epistemic markers which are usually verbs marking the cognitive awareness of emotions in the complement position. The epistemic markers include verbs of seeing, hearing, knowing, and existing.

5.2 Linguistic Patterns

With the six groups of linguistic cues, we generalize 14 rules used in Lee et al. (2010b) to locate the clause positions of an emotion cause, as shown in Table 4. The abbreviations used in the rules are given as follows:

C = Cause
K = Emotion keyword
B = Clauses before the focus clause
F = Focus clause/the clause containing the emotion verb
A = Clauses after the focus clause

Table 4: Linguistic rules for emotion cause detection (Lee et al. 2010b)

No.	Rules
1	i) C(B/F) + III(F) + K(F) ii) C = the nearest N/V before I in F/B
2	i) IV/V/I/II(B/F) + C(B/F) + K(F) ii) C = the nearest N/V before K in F
3	i) I/II/IV/V (B) + C(B) + K(F) ii) C = the nearest N/V after I/II/IV/V in B
4	i) K(F) + V/VI(F) + C(F/A) ii) C = the nearest N/V after V/VI in F/A
5	i) K(F)+II(A)+C(A) ii) C = the nearest N/V after II in A
6	i) III(F) + K(F) + C(F/A) ii) C = the nearest N/V after K in F or A
7	i) *yue4* C *yue4* K "the more C the more K" (F) ii) C = the V in between the two *yue4*'s in F
8	i) K(F) + C(F) ii) C = the nearest N/V after K in F
9	i) V(F) + K(F) ii) C = V+(an aspectual marker) in F
10	i) K(F) + *de* "possession"(F) + C(F) ii) C = the nearest N/V +的+N after *de* in F
12	i) K(B) + IV (B) + C(F) ii) C = the nearest N/V after IV in F
13	i) IV(B) + C(B) + K(F) ii) C = the nearest N/V after IV in B
14	i) C(B) + K(F) ii) C = the nearest N/V before K in B

For illustration, an example of the rule description is given in Rule 1.

Rule 1:
i) C(B/F) + III(F) + K(F)
ii) C = the nearest N/V before III in F/B

Rule 1 indicates that the cause (C) comes before Group III cue words. Theoretically, in identifying C, we look for the nearest verb/noun occurring before Group III cue words in the focus clause (F) or the clauses before the focus clause (B), and consider the clause containing this verb/noun as a cause. Practically, for each cause candidate, i.e. 'left_1', if it contains this verb/noun, we create a feature with "left_1_rule_1=1".

5.3 Generalized Patterns

Rule-based patterns usually achieve a rather high accuracy, but suffer from low coverage. To avoid this shortcoming, we extract a generalized feature automatically according to the rules in Table 4. The features are able to detect two kinds of constructions, namely functional constructions, i.e. rhetorical constructions, and specific constructions for emotion causes.

Local functional constructions: a cause occurring in the focus clause is often expressed with certain functional words, such as "*because of*", "*due to*". In order to capture the various expressions of these functional constructions, we identify all functional words around the given emotion keyword. For an emotion keyword, we search 'left_0' from the right until a noun or a verb is found. Next, all unigrams and bigrams between the noun or the verb and the emotion keyword are extracted. The same applies to 'right_0'.

Long-distance conjunction constructions: Group II enumerates only some typical conjunction words. To capture more general rhetorical relations, according to the given POS tags, the conjunction word is extracted for each cause candidate, if it occurs at the beginning of the candidate.

Generalized action and epistemic verbs: Group IV and V cover only partial action and epistemic verbs. To capture possible related expressions, we take the advantage of Chinese characters. In Chinese, each character itself usually has a meaning and some characters have a strong capability to create words with extended meaning. For example, the character "*ting1*-listen" combines with other characters to create words expressing "listening", such as *ting1jian4*, *ting1wen5*. With the selected characters regarding reported verbs and epistemic markers, each cause candidate is checked to see whether it contains the predefined characters.

6 Experiments

For the emotion cause corpus, we reserve 80% as the training data, 10% as the development data, and 10% as the test data. During evaluation, we first convert the multi-label tag outputted from our system into a binary tag ('Y' means the presence of a causal relation; 'N' indicates the absence of a causal relation) between the emotion keyword and each candidate in its corresponding cause candidates. Thus, the evaluation scores for binary classification based on three common measures, i.e. precision, recall and F-score, are chosen.

6.1 Linguistic Feature Analysis

According to the distribution in Table 1, we design a naive baseline to allow feature analysis. The baseline searches for the cause candidates in the order of <left_1, right_0, left_2, left_0, right_1>. If the candidate contains a noun or verb, consider this clause as a cause and stop.

We run the multi-label system with different groups of features and the performances are shown in Table 5. The feature set begins with linguistic patterns (LP), and is then incorporated with local functional constructions (LFC), long-distance conjunction constructions (LCC), and generalized action and epistemic verbs (GAE), one by one. Since the 'N' tag is overwhelming, we report only the Mac average scores for both 'Y' and 'N' tags.

In Table 5, we first notice that the performances achieve significant improvement from the baseline to the final system (~17%). This indicates that our linguistic features are effective for emotion cause detection. In addition, we observe that LP and LFC are the best two effective features, whereas LCC and GAE have slight contributions. This shows that our feature extraction has a strong capability to detect local causal constructions, and is yet unable to detect the long-distance or semantic causal information. Here, 'local' refers to the information in the focus clause. We also find that incorporating LFC, which is a pure local feature, generally improves the performances of all cause candidates, i.e. ~5% improvement for 'left_1'. This indicates that our multi-label integrated system is able to convey information among cause candidates.

Table 5: The overall performance with different feature sets of the multi-label system

	Precision	Recall	F-score
Baseline	56.64	57.70	56.96
LP	74.92	66.70	69.21
+ LFC	72.80	71.94	72.35
+ LCC	73.60	72.50	73.02
+ GAE	73.90	72.70	73.26

Table 6: The separate performances for 'Y' and 'N' tags of the multi-label system

	'Y'	'N'
Baseline	33.06	80.85
LP	48.32	90.11
+ LFC	55.45	89.24
+ LCC	56.48	89.57
+ GPE	56.84	89.68

Table 6 shows the performances (F-scores) for 'Y' and 'N' tags separately. First, we notice that the performances of the 'N' tag are much better than the ones of 'Y' tag. Second, it is surprising that incorporating the linguistic features significantly improves only the 'Y' tag (from 33% to 56%), but does not affect 'N' tag. This suggests that our linguistic features are effective to detect the presence of causal relation, and yet do not hurt the detection of 'non_causal' relation. For the 'Y' tag, the features LP and LFC achieve ~15% and ~7% improvements respectively. LCC and GPE, on the other hand, show slight improvements only.

Finally, Table 7 shows the detailed performances of our multi-label system with all features. The last row shows the overall performances of 'Y' and 'N' tags. For the 'Y' tag, the closer the cause candidates are to the emotion keyword, the better performances the system achieves. This proves that the features we propose effectively detect local emotion causes, more effort,

Table 7: The detailed performance for the multi-label system including all features

'Y' tag	Precision	Recall	F-score	'N' tag	Precision	Recall	F-score
Left_0	68.92	68.92	68.92	Left_0	93.72	93.72	93.72
Left_1	57.63	63.35	60.36	Left_1	82.90	79.22	81.02
Left_2	29.27	20.69	24.24	Left_2	89.23	92.93	91.04
Right_0	67.78	64.89	66.30	Right_0	82.63	84.41	83.51
Right_1	54.84	30.91	39.54	Right_1	92.00	96.90	94.38
Total	58.84	54.98	56.84	Total	88.96	90.42	89.68

Table 8: The detailed performance for the single-label system including all features

'Y' tag	Precision	Recall	F-score	'N' tag	Precision	Recall	F-score
Left_0	65.39	68.92	67.11	Left_0	93.65	92.62	93.13
Left_1	61.19	50.93	55.59	Left_1	79.64	85.60	82.51
Left_2	28.57	20.69	24.00	Left_2	89.20	92.68	90.91
Right_0	70.13	57.45	63.16	Right_0	80.30	87.63	83.81
Right_1	33.33	40.00	36.36	Right_1	92.50	90.24	91.36
Total	55.67	50.00	52.68	Total	87.85	90.08	88.95

however, should be put on the detection of long-distance causes. In addition, we find that the detection of long-distance causes usually relies on two kinds of information for inference: rhetorical relation and deep semantic information.

6.2 Modeling Analysis

To compare our multi-label model with single-label models, we create a single-label system as follows. The single-label model is a binary classification for a pair comprising the emotion keyword and a candidate in its corresponding cause candidates. For each pair, all linguistic features are extracted only from the focus clause and its corresponding cause candidate. Note that we only use the features in the focus clause for "left_0" and "right_0". The performances are shown in Table 8.

Comparing Tables 7 and 8, all F-scores of the 'Y' tag increase and the performances of the 'N' tag remain almost the same for both the single-label model and our multi-label model. We also find that the multi-label model takes more advantage of local information, and improves the performances, particularly for "left_1".

To take an in-depth analysis of the cause detection capability of the multi-label model, an evaluation is designed that the label is treated as a tag from the multi-label classifier. Due to the tag sparseness problem (as in Table 2), only the "left_2, left_1" tag is detected in the test data, and its performance is 21% precision, 26% recall and 23% F-score. Furthermore, we notice that ~18% of the "left_1" tags are detected through this combination tag. This shows that some causes need to take into account the mutual information between clauses. Although the scores are low, it still shows that our multi-label model provides an effective way of detecting some of the multi-clauses causes.

7 Conclusion

We treat emotion cause detection as a multi-label task, and develop two sets of linguistic features for emotion cause detection based on linguistic cues. The experiments on the small-scale corpus show that both the multi-label model and the linguistic features are able to effectively detect emotion causes. The automatic detection of emotion cause will in turn allow us to extract directly relevant information for public opinion mining and event prediction. It can also be used to improve emotion detection and classification. In the future, we will attempt to improve our system from two aspects. On the one hand, we will explore more powerful multi-label classification models for our system. On the other hand, we will investigate more linguistic patterns or semantic information to further help emotion cause detection.

References

Abbasi, A., H. Chen, S. Thoms, and T. Fu. 2008. Affect Analysis of Web Forums and Blogs using Correlation Ensembles". In *IEEE Tran. Knowledge and Data Engineering*, vol. 20(9), pp. 1168-1180.

Bethard, S. and J. Martin. 2008. Learning Semantic Links from a Corpus of Parallel Temporal and Causal Relations. In *Proceedings of ACL.*

Descartes, R. 1649. The Passions of the Soul. In J. Cottingham et al. (Eds), *The Philosophical Writings of Descartes*. Vol. 1: 325-404.

Chang, D.-S. and K.-S. Choi. 2006. Incremental cue phrase learning and bootstrapping method for causality extraction using cue phrase and word pair probabilities. *Information Processing and Management*. 42(3): 662-678.

Chen, Y., S. Y. M. Lee and C.-R. Huang. 2009. Are Emotions Enumerable or Decomposable? And Its Implications for Emotion Processing. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*.

Girju, R. 2003. Automatic Detection of Causal Relations for Question Answering. In *the 41st Annual Meeting of the Association for Computational Linguistics, Workshop on Multilingual Summarization and Question Answering - Machine Learning and Beyond*, Sapporo, Japan.

James, W. 1884. What is an Emotion? *Mind*, 9(34):188–205.

Lee, S. Y. M., Y. Chen and C.-R. Huang. 2010a. A Text-driven Rule-based System for Emotion Cause Detection. In *Proceedings of NAACL-HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*.

Lee, S. Y. M., Y. Chen, S. Li and C.-R. Huang. 2010b. Emotion Cause Events: Corpus Construction and Analysis. In *Proceedings of LREC 2010*.

Low, B. T., K. Chan , L. L. Choi , M. Y. Chin , S. L. Lay. 2001. Semantic Expectation-Based Causation Knowledge Extraction: A Study on Hong Kong Stock Movement Analysis, In *Proceedings of the 5th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, p.114-123, April 16-18.

Marcu, D., and A. Echihabi. 2002. An Unsupervised Approach to Recognizing Discourse Relations. In *Proceedings of ACL.*

Mihalcea, R. and H. Liu. 2006. A Corpus-based Approach to Finding Happiness. In *Proceedings of the AAAI Spring Symposium on Computational Approaches to Weblogs*.

Persing, I. and V. Ng. 2009. Semi-Supervised Cause Identification from Aviation Safety Reports. In *Proceedings of ACL.*

Plutchik, R. 1980. *Emotions: A Psychoevolutionary Synthesis*. New York: Harper & Row.

Strapparava, C. and R. Mihalcea. 2008. Learning to Identify Emotions in Text. In *Proceedings of the ACM Conference on Applied Computing ACM-SAC*.

Tokuhisa, R., K. Inui, and Y. Matsumoto. 2008. Emotion recognition Using Massive Examples Extracted from the Web. In *Proceedings of COLING.*

Wierzbicka, A. 1999. *Emotions across Languages and Cultures: Diversity and Universals*. Cambridge: Cambridge University Press.

A Twin-Candidate Based Approach for Event Pronoun Resolution using Composite Kernel

Chen Bin[1] **Su Jian**[2] **Tan Chew Lim**[1]

[1]National University of Singapore
{chenbin,tancl}@comp.nus.edu.sg

[2]Institute for Inforcomm Research, A-STAR
sujian@i2r.a-star.edu.sg

Abstract

Event Anaphora Resolution is an important task for cascaded event template extraction and other NLP study. In this paper, we provide a first systematic study of resolving pronouns to their event verb antecedents for general purpose. First, we explore various positional, lexical and syntactic features useful for the event pronoun resolution. We further explore tree kernel to model structural information embedded in syntactic parses. A composite kernel is then used to combine the above diverse information. In addition, we employed a twin-candidate based preferences learning model to capture the pair wise candidates' preference knowledge. Besides we also look into the incorporation of the negative training instances with anaphoric pronouns whose antecedents are not verbs. Although these negative training instances are not used in previous study on anaphora resolution, our study shows that they are very useful for the final resolution through random sampling strategy. Our experiments demonstrate that it's meaningful to keep certain training data as development data to help SVM select a more accurate hyper plane which provides significant improvement over the default setting with all training data.

1 Introduction

Anaphora resolution, the task of resolving a given text expression to its referred expression in prior texts, is important for intelligent text processing systems. Most previous works on anaphora resolution mainly aims at object anaphora in which both the anaphor and its antecedent are mentions of the same real world objects

In contrast, an event anaphora as first defined in (Asher, 1993) is an anaphoric reference to an event, fact, and proposition which is representative of eventuality and abstract entity. Consider the following example:

This was an all-white, all-Christian community that all the sudden was taken over -- not taken over, that's a very bad choice of words, but **[*invaded*]$_1$** *by, perhaps different groups.*
[*It*]$_2$ *began when a Hasidic Jewish family bought one of the town's two meat-packing plants 13 years ago.*

The anaphor **[*It*]$_2$** in the above example refers back to an event, "all-white and all-Christian city of Postville is diluted by different ethnic groups." Here, we take the main verb of the event, **[*invaded*]$_1$** as the representation of this event and the antecedent for pronoun **[*It*]$_2$**.

According to (Asher, 1993), antecedents of event pronoun include both gerunds (e.g. destruction) and inflectional verbs (e.g. destroying). In our study, we focus on the inflectional verb representation, as the gerund representation is studied in the conventional anaphora resolution. For the rest of this paper, "event pronouns" are pronouns whose antecedents are event verbs while "non-event anaphoric pronouns" are those with antecedents other than event verbs.

Entity anaphora resolution provides critical links for cascaded event template extraction. It also provides useful information for further inference needed in other natural language processing tasks such as discourse relation and entailment. Event anaphora (both pronouns and noun phrases) contributes a significant proportion in anaphora corpora, such as OntoNotes. 19.97% of its total number of entity chains contains event verb mentions.

In (Asher, 1993) chapter 6, a method to resolve references to abstract entities using discourse representation theory is discussed. However, no computation system was proposed for entity anaphora resolution. (Byron, 2002) proposed semantic filtering as a complement to salience calculations to resolve event pronoun targeted by us. This knowledge deep approach only

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 188–196,
Beijing, August 2010

works for much focused domain like trains spoken dialogue with handcraft knowledge of relevant events for only limited number of verbs involved. Clearly, this approach is not suitable for general event pronoun resolution say in news articles. Besides, there's also no specific performance report on event pronoun resolution, thus it's not clear how effective their approach is. (Müller, 2007) proposed pronoun resolution system using a set of hand-crafted constraints such as "argumenthood" and "right-frontier condition" together with logistic regression model based on corpus counts. The event pronouns are resolved together with object pronouns. This explorative work produced an 11.94% F-score for event pronoun resolution which demonstrated the difficulty of event anaphora resolution. In (Pradhan, *et.al*, 2007), a general anaphora resolution system is applied to OntoNotes corpus. However, their set of features is designed for object anaphora resolution. There is no specific performance reported on event anaphora. We suspect the event pronouns are not correctly resolved in general as most of these features are irrelevant to event pronoun resolution.

In this paper, we provide the first systematic study on pronominal references to event antecedents. First, we explore various positional, lexical and syntactic features useful for event pronoun resolution, which turns out quite different from conventional pronoun resolution except sentence distance information. These have been used together with syntactic structural information using a composite kernel. Furthermore, we also consider candidates' preferences information using twin-candidate model.

Besides we further look into the incorporation of negative instances from non-event anaphoric pronoun, although these instances are not used in previous study on co-reference or anaphora resolution as they make training instances extremely unbalanced. Our study shows that they can be very useful for the final resolution after random sampling strategy.

We further demonstrate that it's meaningful to keep certain training data as development data to help SVM select a more accurate hyper-plane which provide significant improvement over the default setting with all training data.

The rest of this paper is organized as follows. Section 2 introduces the framework for event pronoun resolution, the considerations on training instance, the various features useful for event pronoun resolution and SVM classifier with adjustment of hyper-plane. Twin-candidate model is further introduced to capture the preferences among candidates. Section 3 presents in details the structural syntactic feature and the kernel functions to incorporate such a feature in the resolution. Section 4 presents the experiment results and some discussion. Section 5 concludes the paper.

2 The Resolution Framework

Our event-anaphora resolution system adopts the common learning-based model for object anaphora resolution, as employed by (Soon *et al.*, 2001) and (Ng and Cardie, 2002a).

2.1 Training and Testing instance

In the learning framework, training or testing instance of the resolution system has a form of $fv(candi_i, ana)$ where $candi_i$ is the i^{th} candidate of the antecedent of anaphor ana. An instance is labeled as positive if $candi_i$ is the antecedent of ana, or negative if $candi_i$ is not the antecedent of ana. An instance is associated with a feature vector which records different properties and relations between ana and $candi_i$. The features used in our system will be discussed later in this paper.

During training, for each event pronoun, we consider the preceding verbs in its current and previous two sentences as its antecedent candidates. A positive instance is formed by pairing an anaphor with its correct antecedent. And a set of negative instances is formed by pairing an anaphor with its candidates other than the correct antecedent. In addition, more negative instances are generated from non-event anaphoric pronouns. Such an instance is created by pairing up a non-event anaphoric pronoun with each of the verbs within the pronoun's sentence and previous two sentences. This set of instances from non-event anaphoric pronouns is employed to provide extra power on ruling out non-event anaphoric pronouns during resolution. This is inspired by the fact that event pronouns are only 14.7% of all the pronouns in the OntoNotes corpus. Based on these generated training instances, we can train a binary classifier using any discriminative learning algorithm.

The natural distribution of textual data is often imbalanced. Classes with fewer examples are under-represented and classifiers often perform far below satisfactory. In our study, this becomes a significant issue as positive class (event anaphoric) is the minority class in pronoun resolution task. Thus we utilize a random down sampling method to reduce majority class samples to an equivalent level with the minority class samples which is described in (Kubat and Matwin, 1997) and (Estabrooks *et al*, 2004). In (Ng and Cardie, 2002b), they proposed a negative sample selection scheme which included only negative instances found in between an anaphor and its antecedent. However, in our event pronoun resolution, we are distinguishing the event-anaphoric from non-event anaphoric which is different from (Ng and Cardie, 2002b).

2.2 Feature Space

In a conventional pronoun resolution, a set of syntactic and semantic knowledge has been reported as in (Strube and Müller, 2003; Yang *et al*, 2004;2005a;2006). These features include number agreement, gender agreement and many others. However, most of these features are not useful for our task, as our antecedents are inflectional verbs instead of noun phrases. Thus we have conducted a study on effectiveness of potential positional, lexical and syntactic features. The lexical knowledge is mainly collected from corpus statistics. The syntactic features are mainly from intuitions. These features are purposely engineered to be highly correlated with positive instances. Therefore such kind of features will contribute to a high precision classifier.

- **Sentence Distance**

This feature measures the sentence distance between an anaphor and its antecedent candidate under the assumptions that a candidate in the closer sentence to the anaphor is preferred to be the antecedent.

- **Word Distance**

This feature measures the word distance between an anaphor and its antecedent candidate. It is mainly to distinguish verbs from the same sentence.

- **Surrounding Words and POS Tags**

The intuition behind this set of features is to find potential surface words that occur most frequently with the positive instances. Since most of verbs occurred in front of pronoun, we have built a frequency table from the preceding 5 words of the verb to succeeding 5 surface words of the pronoun. After the frequency table is built, we select those words with confidence[1] > 70% as features. Similar to Surrounding Words, we have built a frequency table to select indicative surrounding POS tags which occurs most frequently with positive instances.

- **Co-occurrences of Surrounding Words**

The intuition behind this set of features is to capture potential surface patterns such as "*It caused…*" and *"It leads to"*. These patterns are associated with strong indication that pronoun "*it*" is an event pronoun. The range for the co-occurrences is from preceding 5 words to succeeding 5 words. All possible combinations of word positions are used for a co-occurrence words pattern. For example "*it leads to*" will generate a pattern as "***S1_S2_lead_to***" where ***S1*** and ***S2*** mean succeeding position 1 and 2. Similar to previous surrounding words, we will conduct corpus statistics analysis and select co-occurrence patterns with a confidence greater than 70%. Following the same process, we have examined co-occurrence patterns for surrounding POS tags.

- **Subject/Object Features**

This set of features aims to capture the relative position of the pronoun in a sentence. It denotes the preference of pronoun's position at the clause level. There are 4 features in this category as listed below.

Subject of Main Clause

This feature indicates whether a pronoun is at the subject position of a main clause.

Subject of Sub-clause

This feature indicates whether a pronoun is at the subject position of a sub-clause.

Object of Main Clause

This feature indicates whether a pronoun is at the object position of a main clause.

Object of Sub-clause

This feature indicates whether a pronoun is at the object position of a sub-clause.

- **Verb of Main/Sub Clause**

Similar to the Subject/Object features of pronoun, the following two features capture the rela-

[1] $Confidence = \frac{\#\ of\ word_i occurred\ with\ positive\ instance}{\#\ of\ word_i occurrences}$

tive position of a verb in a sentence. It encodes the preference of verb position between main verbs in main/sub clauses.

Main Verb in Main Clause

This feature indicates whether a verb is a main verb in a main clause.

Main Verb in Sub-clause

This feature indicates whether a verb is a main verb in a sub-clause.

2.3 Support Vector Machine

In theory, any discriminative learning algorithm is applicable to learn a classifier for pronoun resolution. In our study, we use Support Vector Machine (Vapnik, 1995) to allow the use of kernels to incorporate the structure feature. One advantage of SVM is that we can use tree kernel approach to capture syntactic parse tree information in a particular high-dimension space.

Suppose a training set S consists of labeled vectors $\{(x_i, y_i)\}$, where x_i is the feature vector of a training instance and y_i is its class label. The classifier learned by SVM is:

$$f(x) = sign\left(\sum_{i=1} y_i a_i x \cdot x_i + b\right)$$

where a_i is the learned parameter for a support vector x_i. An instance x is classified as positive if $f(x) \geq 0$. Otherwise, x is negative.

- **Adjust Hyper-plane with Development Data**

Previous works on pronoun resolution such as (Yang *et al*, 2006) used the default setting for hyper-plane which sets $\mathrm{f(x)} = 0$. And an instance is positive if $f(x) \geq 0$ and negative otherwise. In our study, we look into a method of adjusting the hyper-plane's position using development data to improve the classifier's performance.

Considering a default model setting for SVM as shown in Figure 2(for illustration purpose, we use a 2-D example).

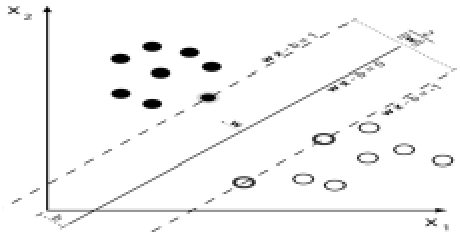


Figure 2: 2-D SVM Illustration

The objective of SVM learning process is to find a set of weight vector w which maximizes the margin (defined as $\frac{2}{\|w\|}$) with constraints defined by support vectors. The separating hyper-plane is given by $w \cdot x + b = 0$ as bold line in the center. The margin is the region between the two dotted lines (bounded by $w \cdot x + b = 1$ and $w \cdot x + b = -1$). The margin is a space without any information from training instances. The actual hyper-plane may fall in any place within the margin. It does not necessarily occur in the. However, the hyper-plane is used to separate positive and negative instances during classification process without consideration of the margin. Thus if an instance falls in the margin, SVM can only decide class label from hyper-plane which may cause misclassification in the margin.

Based on the previous discussion, we propose an adjustment of the hyper-plane using development data. For simplicity, we adjust the hyper-plane function value instead of modeling the function itself. The hyper-plane function value will be further referred as a threshold θ. The following is a modified version of a learned SVM classifier.

$$f(x,\theta) = \begin{cases} 1 & \text{if} \left(\sum_{i=1} y_i a_i x \cdot x_i + b\right) \geq \theta \\ -1 & \text{if} \left(\sum_{i=1} y_i a_i x \cdot x_i + b\right) < \theta \end{cases}$$

where θ is the threshold, a_i is the learned parameter for a feature x_i and y_i is its class label. A set of development data is used to adjust the hyper-plane function threshold θ in order to maximize the accuracy of the learned SVM classifier on development data. The adjustment of hyper-plane is defined as:

$$\theta_{best} = argmax_{\theta \in \Theta}(\sum_{x \in X} I(y, f(x,\theta)))$$

where $I(y,f)$ is an indicator function which output 1 if $f(x,\theta)$is same sign as y and 0 otherwise. Thereafter, the learned threshold θ is applied to the testing set.

3 Incorporating Structural Syntactic Information

A parse tree that covers a pronoun and its antecedent candidate could provide us much syntactic information related to the pair which is explicitly or implicitly represented in the tree. Therefore, by comparing the common sub-structures between two trees we can find out to what degree two trees contain similar syntactic information, which can be done using a convolution tree kernel. The value returned from tree kernel reflects similarity between two instances in syntax. Such syntactic similarity can be further combined with other knowledge to compute overall similarity between two instances, through a composite kernel. Normally, parsing is done at sentence level. However, in many cases a pronoun and its antecedent candidate do not occur in the same sentence. To present their syntactic properties and relations in a single tree structure, we construct a syntax tree for an entire text, by attaching the parse trees of all its sentences to an upper node. Having obtained the parse tree of a text, we shall consider how to select the appropriate portion of the tree as the structured feature for a given instance. As each instance is related to a pronoun and a candidate, the structured feature at least should be able to cover both of these two expressions.

3.1 Structural Syntactic Feature

Generally, the more substructure of the tree is included, the more syntactic information would be provided, but at the same time the more noisy information that comes from parsing errors would likely be introduced. In our study, we examine three possible structured features that contain different substructures of the parse tree:

- **Minimum Expansion Tree**

This feature records the minimal structure covering both pronoun and its candidate in parse tree. It only includes the nodes occurring in the shortest path connecting the pronoun and its candidate, via the nearest commonly commanding node. When the pronoun and candidate are from different sentences, we will find a path through pseudo "TOP" node which links all the parse trees. Considering the example given in section 1,

This was an all-white, all-Christian community that all the sudden was taken over -- not taken over, that's a very bad choice of words, but **[*invaded*]$_1$** *by, perhaps different groups.*

[*It*]$_2$ *began when a Hasidic Jewish family bought one of the town's two meat-packing plants 13 years ago.*

The minimum expansion structural feature of the instance {***invaded, it***} is annotated with bold lines and shaded nodes in figure 1.

- **Simple Expansion Tree**

Minimum-Expansion could, to some degree, describe the syntactic relationships between the candidate and pronoun. However, it is incapable of capturing the syntactic properties of the candidate or the pronoun, because the tree structure surrounding the expression is not taken into consideration. To incorporate such information, feature Simple-Expansion not only contains all the nodes in Minimum-Expansion, but also includes the first-level children of these nodes[2] except the punctuations. The simple-expansion structural feature of instance **{*invaded, it*}** is annotated in figure 2. In the left sentence's tree, the node "NP" for "*perhaps different groups*" is terminated to provide a clue that we have a noun phrase at the object position of the candidate verb.

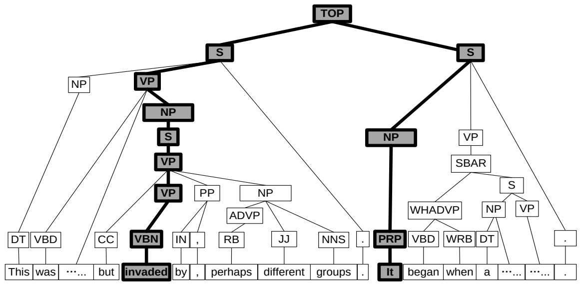


Figure 1: Minimum-Expansion Tree

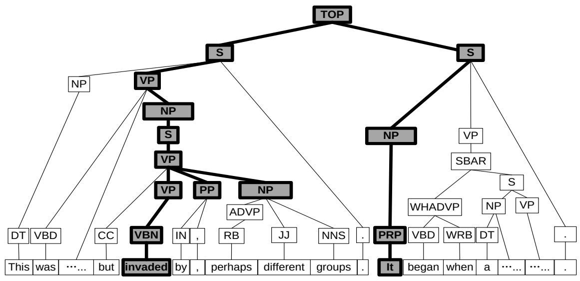


Figure 2: Simple Expansion Tree

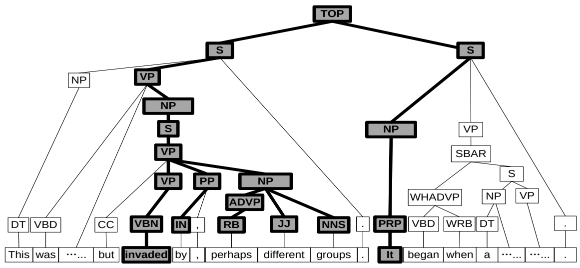


Figure 3: Full-Expansion Tree

- **Full Expansion Tree**

This feature focuses on the whole tree structure between the candidate and pronoun. It not only includes all the nodes in Simple-Expansion, but also the nodes (beneath the nearest commanding parent) that cover the words between the candi-

[2] If the pronoun and the candidate are not in the same sentence, we will not include the nodes denoting the sentences before the candidate or after the pronoun.

date and the pronoun[3]. Such a feature keeps the most information related to the pronoun and candidate pair. Figure 3 shows the structure for feature full-expansion for instance {***invaded, it***}. As illustrated, the "NP" node for **"***perhaps different groups***"** is further expanded to the POS level. All its child nodes are included in the full-expansion tree except the surface words.

3.2 Convolution Parse Tree Kernel and Composite Kernel

To calculate the similarity between two structured features, we use the convolution tree kernel that is defined by Collins and Duffy (2002) and Moschitti (2004). Given two trees, the kernel will enumerate all their sub-trees and use the number of common sub-trees as the measure of similarity between two trees. The above tree kernel only aims for the structured feature. We also need a composite kernel to combine the structured feature and the flat features from section 2.2. In our study we define the composite kernel as follows:

$$K_{comp}(x_1, x_2) = \frac{K_{tree}(x_1, x_2)}{|K_{tree}(x_1, x_2)|} + \frac{K_{flat}(x_1, x_2)}{|K_{flat}(x_1, x_2)|}$$

where K_{tree} is the convolution tree kernel defined for the structured feature, and K_{flat} is the kernel applied on the flat features. Both kernels are divided by their respective length[4] for normalization. The new composite kernel K_{comp}, defined as the sum of normalized K_{tree} and K_{flat}, will return a value close to 1 only if both the structured features and the flat features have high similarity under their respective kernels.

3.3 Twin-Candidate Framework using Ranking SVM Model

In a ranking SVM kernel as described in (Moschitti *et al, 2006*) for Semantic Role Labeling, two argument annotations (as argument trees) are presented to the ranking SVM model to decide which one is better. In our case, we present two syntactic trees from two candidates to the ranking SVM model. The idea is inspired by (Yang, *et.al*, 2005b;2008). The intuition behind the twin-candidate model is to capture the information of how much one candidate is more preferred than another. The candidate wins most of the pair wise comparisons is selected as antecedent.

The feature vector for each training instance has a form of $fv = (candi_i, candi_j)$. An instance is positive if $cand_i$ is a better antecedent choice than $candi_j$. Otherwise, it is a negative instance. For each feature vector, both tree structural features and flat features are used. Thus each feature vector has a form of $fv = (t_i, t_j, v_i, v_j)$ where t_i and t_j are trees of candidate i and j respectively, v_i and v_j are flat feature vectors of candidate i and j respectively.

In the training instances generation, we only generate those instances with one candidate is the correct antecedent. This follows the same strategy used in (Yang *et al*, 2008) for object anaphora resolution.

In the resolution process, a list of m candidates is extracted from a three sentences window. A total of $\binom{m}{2}$ instances are generated by pairing-up the m candidates pair-wisely. We used a Round-Robin scoring scheme for antecedent selection. Suppose a SVM output for an instance $fv = (candi_i, candi_j)$ is 1, we will give a score 1 for $candi_i$ and -1 for $candi_j$ and vice versa. At last, the candidate with the highest score is selected as antecedent. In order to handle a non-event anaphoric pronoun, we have set a threshold to distinguish event anaphoric from non-event anaphoric. A pronoun is considered as event anaphoric if its score is above the threshold. In our experiments, we kept a set of development data to find out the threshold in an empirical way.

4 Experiments and Discussions

4.1 Experimental Setup

OntoNotes Release 2.0 English corpus as in (Hovy *et al*, 2006) is used in our study, which contains 300k words of English newswire data (from the Wall Street Journal) and 200k words of English broadcast news data (from ABC, CNN, NBC, Public Radio International and Voice of America). Table 1 shows the distribution of various entities. We focused on the resolution of 502 event pronouns encountered in the corpus. The resolution system has to handle both the event pronoun identification and antecedent selection tasks. To illustrate the difficulty of event pronoun resolution, 14.7% of all pronoun mentions are event anaphoric and only 31.5% of

[3] We will not expand the nodes denoting the sentences other than where the pronoun and the candidate occur.

[4] The length of a kernel K is defined as $|K(x_1, x_2)| = \sqrt{K(x_1, x_1) \cdot K(x_2, x_2)}$

event pronoun can be resolved using "most recent verb" heuristics. Therefore a most-recent-verb baseline will yield an f-score 4.63%.

To conduct event pronoun resolution, an input raw text was preprocessed automatically by a pipeline of NLP components. The noun phrase identification and the predicate-argument extraction were done based on Stanford Parser (Klein and Manning, 2003a;b) with F-score of 86.32% on Penn Treebank corpus.

<table>
<tr><td colspan="3">Non-Event Anaphora: 4952 80.03%</td></tr>
<tr><td rowspan="4">Event Anaphora:
1235
19.97%</td><td colspan="2">Event NP: 733 59.35%</td></tr>
<tr><td rowspan="3">Event Pronoun:
502 40.65%</td><td>It: 29.0%</td></tr>
<tr><td>This: 16.9%</td></tr>
<tr><td>That: 54.1%</td></tr>
</table>

Table 1: The distribution of various types of 6187 anaphora in OntoNotes 2.0

For each pronoun encountered during resolution, all the inflectional verbs within the current and previous two sentences are taken as candidates. For the current sentence, we take only those verbs in front of the pronoun. On average, each event pronoun has 6.93 candidates. Non-event anaphoric pronouns will generate 7.3 negative instances on average.

4.2 Experiment Results and Discussion

In this section, we will present our experimental results with discussions. The performance measures we used are precision, recall and F-score. All the experiments are done with a 10-folds cross validation. In each fold of experiments, the whole corpus is divided into 10 equal sized portions. One of them is selected as testing corpus while the remaining 9 are used for training. In experiments with development data, 1 of the 9 training portions is kept for development purpose. In case of statistical significance test for differences is needed, a two-tailed, paired-sample Student's t-Test is performed at 0.05 level of significance.

In the first set of experiments, we are aiming to investigate the effectiveness of each single knowledge source. Table 2 reports the performance of each individual experiment. The flat feature set yields a baseline system with 40.6% f-score. By using each tree structure along, we can only achieve a performance of 44.4% f-score using the minimum-expansion tree. Therefore, we will further investigate the different ways of combining flat and syntactic structure knowledge to improve resolution performances.

	Precision	**Recall**	**F-score**
Flat	0.406	0.406	0.406
Min-Exp	0.355	0.596	0.444
Simple-Exp	0.347	0.512	0.414
Full-Exp	0.323	0.476	0.385

Table 2: Contribution from Single Knowledge Source

The second set of experiments is conducted to verify the performances of various tree structures combined with flat features. The performances are reported in table 3. Each experiment is reported with two performances. The upper one is done with default hyper-plane setting. The lower one is done using the hyper-plane adjustment as we discussed in section 2.3.

	Precision	**Recall**	**F-score**
Min-Exp + Flat	0.433 *(0.727)*	0.512 *(0.446)*	0.469 *(0.553)*
Simple-Exp +Flat	0.423 *(0.652)*	0.534 *(0.492)*	0.472 *(0.561)*
Full-Exp + Flat	0.416 *(0.638)*	0.526 *(0.496)*	0.465 *(0.558)*

Table 3: Comparison of Different Tree Structure +Flat

As table 3 shows, minimum-expansion gives highest precision in both experiment settings. Minimum-expansion emphasizes syntactic structures linking the anaphor and antecedent. Although using only the syntactic path may lose the contextual information, but it also prune out the potential noise within the contextual structures. In contrast, the full-expansion gives the highest recall. This is probably due to the widest knowledge coverage provides by the full-expansion syntactic tree. As a trade-off, the precision of full-expansion is the lowest in the experiments. One reason for this may be due to OntoNotes corpus is from broadcasting news domain. Its texts are less-formally structured. Another type of noise is that a narrator of news may read an abnormally long sentence. It should appear as several separate sentences in a news article. However, in broadcasting news, these sentences maybe simply joined by conjunction word "and". Thus a very nasty and noisy structure is created from it. Comparing the three knowledge source, simple-expansion achieves moderate precision and recall which results in the highest f-score. From this, we can draw a conclusion that simple-expansion achieves a balance between the indicative structural information and introduced noises.

In the next set of experiments, we will compare different setting for training instances generation. A typical setting contains no negative instances generated from non-event anaphoric pronoun. This is not an issue for object pronoun resolution as majority of pronouns in an article is anaphoric. However in our case, the event pronoun consists of only 14.7% of the total pronouns in OntoNotes. Thus we incorporate the instances from non-event pronouns to improve the precision of the classifier. However, if we include all the negative instances from non-event anaphoric pronouns, the positive instances will be overwhelmed by the negative instances. A down sampling is applied to the training instances to create a more balanced class distribution. Table 4 reports various training settings using simple-expansion tree structure.

Simple-Exp Tree	Precision	Recall	F-score
Without Non-event Negative	0.423	**0.534**	0.472
Incl. All Negative	**0.733**	0.410	0.526
Balanced Negative	0.599	0.506	0.549
Development Data	0.652	0.492	**0.561**

Table 4: Comparison of Training Setup, Simple-Exp

In table 4, the first line is experiment without any negative instances from non-event pronouns. The second line is the performance with all negative instances from non-event pronouns. Third line is performance using a balanced training set using down sampling. The last line is experiment using hyper-plane adjustment. The first line gives the highest recall measure because it has no discriminative knowledge on non-event anaphoric pronoun. The second line yields the highest precision which complies with our claim that including negative instances from non-event pronouns will improve precision of the classifier because more discriminative power is given by non-event pronoun instances. The balanced training set achieves a better f-score comparing to models with no/all negative instances. This is because balanced training set provides a better weighted positive/negative instances which implies a balanced positive/negative knowledge representation. As a result of that, we achieve a better balanced f-score. In (Ng and Cardie, 2002b), they concluded that only the negative instances in between the anaphor and antecedent are useful in the resolution. It is same as our strategy without negative instances from non-event anaphoric pronouns. However, our study showed an improvement by adding in negative instances from non-event anaphoric pronouns as showed in table 4. This is probably due to our random sampling strategy over the negative instances near to the event anaphoric instances. It empowers the system with more discriminative power. The best performance is given by the hyper-plane adaptation model. Although the number of training instances is further reduced for development data, we can have an adjustment of the hyper-plane which is more fit to dataset.

In the last set of experiments, we will present the performance from the twin-candidates based approach in table 5. The first line is the best performance from single candidate system with hyper-plane adaptation. The second line is performance using the twin-candidates approach.

Simple-Exp Tree	Precision	Recall	F-score
Single Candidate	0.652	0.492	0.561
Twin-Candidates	0.626	**0.540**	**0.579**

Table 5: Single vs. Twin Candidates, Simple-Exp

Comparing to the single candidate model, the recall is significantly improved with a small trade-off in precision. The difference in results is statistically significant using t-test at 5% level of significance. It reinforced our intuition that preferences between two candidates are contributive information sources in co-reference resolution.

5 Conclusion and Future Work

The purpose of this paper is to conduct a systematic study of the event pronoun resolution. We propose a resolution system utilizing a set of flat positional, lexical and syntactic feature and structural syntactic feature. The state-of-arts convolution tree kernel is used to extract indicative structural syntactic knowledge. A twin-candidates preference learning based approach is incorporated to reinforce the resolution system with candidates' preferences knowledge. Last but not least, we also proposed a study of the various incorporations of negative training instances, specially using random sampling to handle the imbalanced data. Development data is also used to select more accurate hyper-plane in SVM for better determination.

To further our research work, we plan to employ more semantic information into the system such as semantic role labels and verb frames.

Acknowledgment

We would like to thank Professor Massimo Poesio from University of Trento for the initial discussion of this work.

References

N. Asher. 1993. *Reference to Abstract Objects in Discourse*. Kluwer Academic Publisher. 1993.

V. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer.1995.

M. Kubat and S. Matwin, 1997. Addressing the curse of imbalanced data set: One sided sampling. In *Proceedings of the Fourteenth International Conference on Machine Learning*,1997. pg179–186.

T. Joachims. 1999. Making large-scale svm learning practical. In *Advances in Kernel Methods - Support Vector Learning*. MIT Press.1999.

W. Soon, H. Ng, and D. Lim. 2001. A machine learning approach to coreference resolution of noun phrases. In *Computational Linguistics*, Vol:27(4), pg521– 544.

D. Byron. 2002. Resolving Pronominal Reference to Abstract Entities, in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*. July 2002. , USA

M. Collins and N. Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*. July 2002. , USA

V. Ng and C. Cardie. 2002a. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*. July 2002. , USA. pg104–111.

V. Ng, and C. Cardie. 2002b. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING02). (2002)*

M. Strube and C. Müller. 2003. A Machine Learning Approach to Pronoun Resolution in Spoken Dialogue. . In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL'03)*, 2003

D. Klein and C. Manning. 2003a. Fast Exact Inference with a Factored Model for Natural Language Parsing. In *Advances in Neural Information Processing Systems 15 (NIPS 2002)*, Cambridge, MA: MIT Press, pp. 3-10.

D. Klein and C.Manning. 2003b. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL'03)*, 2003. pg423-430.

X. Yang, G. Zhou, J. Su, and C.Tan. 2003. Coreference Resolution Using Competition Learning Approach. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL'03)*, 2003. pg176–183.

A. Moschitti. 2004. A study on convolution kernels for shallow semantic parsing. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL'04)*, pg335–342.

A. Estabrooks, T. Jo, and N. Japkowicz. 2004. A multiple resampling method for learning from imbalanced data sets. In *Computational Intelligence V*ol:20(1). pg18–36.

X. Yang, J. Su, G. Zhou, and C. Tan. 2004. Improving pronoun resolution by incorporating coreferential information of candidates. In *Proceedings of 42th Annual Meeting of the Association for Computational Linguistics*, 2004. pg127–134.

X. Yang, J. Su and C.Tan. 2005a. Improving Pronoun Resolution Using Statistics-Based Semantic Compatibility Information. *In Proceedings of Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. June 2005.

X. Yang, J. Su and C.Tan. 2005b. A Twin-Candidates Model for Coreference Resolution with Non-Anaphoric Identification Capability. In *Proceedings of IJCNLP-2005*. Pp. 719-730, 2005

E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel. 2006. OntoNotes: The 90\% Solution. In *Proceedings of the Human Language Technology Conference of the NAACL*, 2006

X. Yang, J. Su and C.Tan. 2006. Kernel-Based Pronoun Resolution with Structured Syntactic Knowledge. In *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL'06)*. July 2006. Australia.

A. Moschitti, Making tree kernels practical for natural language learning. In *Proceedings EACL 2006*, Trento, Italy, 2006.

C. Müller. 2007. Resolving it, this, and that in unrestricted multi-party dialog. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL'07)*. 2007. Czech Republic. pg816–823.

X. Yang, J. Su and C.Tan. 2008. A Twin-Candidates Model for Learning-Based Coreference Resolution. In *Computational Linguistics*, Vol:34(3). pg327-356.

S. Pradhan, L. Ramshaw, R. Weischedel, J. MacBride, and L. Micciulla. 2007. Unrestricted Coreference: Identifying Entities and Events in OntoNotes. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC)*, Sep. 2007.

Unsupervised Synthesis of Multilingual Wikipedia Articles

Chen Yuncong
The Human Language Technology Center
The Hong Kong University of Science and Technology
ee_cyxab@stu.ust.hk

Pascale Fung
The Human Language Technology Center
The Hong Kong University of Science and Technology
pascale@ee.ust.hk

Abstract

In this paper, we propose an unsupervised approach to automatically synthesize Wikipedia articles in multiple languages. Taking an existing high-quality version of any entry as content guideline, we extract keywords from it and use the translated keywords to query the monolingual web of the target language. Candidate excerpts or sentences are selected based on an iterative ranking function and eventually synthesized into a complete article that resembles the reference version closely. 16 English and Chinese articles across 5 domains are evaluated to show that our algorithm is domain-independent. Both subjective evaluations by native Chinese readers and ROUGE-L scores computed with respect to standard reference articles demonstrate that synthesized articles outperform existing Chinese versions or MT texts in both content richness and readability. In practice our method can generate prototype texts for Wikipedia that facilitate later human authoring.

1 Introduction

Wikipedia has over 260 versions in different languages, but the great disparity in their scope and quality is hindering the effective spread of knowledge. The English version is currently the dominant one with over 3 million articles while the Chinese version, for example, has only one tenth the amount. Most Chinese articles suffer from content incoherence and lack of details compared to their English counterparts. Some of these articles are human-authored translation of the English version with varying degrees of accuracy and completeness, and others are ill-arranged combinations of excerpts directly adapted from external sources. The former takes considerable human effort and the latter tends to produce fragmented and incomplete texts. The intuitive solution of machine translation is also not feasible because it hardly provides satisfactory readability.

These problems call for a *synthesis* approach. In order to present the information conveyed by an English article in Chinese, instead of literally translate it, we build a topic-template expressed by the keywords extracted from the English article. Machine-translation of these keywords helps to yield the topic-template in Chinese. Using the topic-template in Chinese, we form a pool of candidate excerpts by retrieving Chinese documents from the Internet. These online documents are usually human-authored and have optimal readability and coherence. Candidate excerpts are further split into segments as synthesis unit. For segment selection, we propose an iterative ranking function that aims to maximize textual similarity, keywords coverage, and content coherence, while penalizes information redundancy.

A feature of our approach is the use of bi-lingual resources throughout the synthesis process. We calculate similarity scores of two texts based on both English and Chinese versions of them, which forms a more precise measure than using either version alone.

For the sake of clarity, we will use English and Chinese as examples of source and target language respectively when describing the methodology. Nonetheless, our approach is not constrained to any specific language pair and supports both direction of synthesis.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 197–205,
Beijing, August 2010

2 Related Work

Much work has been done to explore the multilingualism of Wikipedia. (Adafre et al. 2006) investigated two approaches to identify similarity between articles in different languages for automatic generation of parallel corpus, including a machine-translation based approach and one using a bilingual lexicon derived from the hyperlink structure underlying Wikipedia articles. Both methods rely on pair-wise comparisons made at the sentential level, which hardly account for similarity or coherence in the paragraph scope. Besides it is not a generative algorithm and thus inapplicable to our problem where comparable sentences in Chinese are simply not available.

A generative approach was proposed by (Sauper and Barzilay, 2009) to create highly-structured Wikipedia articles (e.g. descriptions of diseases) composed of information drawn from the Internet. It uses an automatically-induced domain-specific template, and the perceptron algorithm augmented with a global integer linear programming (ILP) formulation to optimize both local fit of information into each section and global coherence across the entire article. This method works only for specific domains where articles have obviously separable sections (e.g. Causes and Symptoms) and it requires a training corpus for each domain to induce the template. Moreover, the synthesis units they use are complete excerpts rather than individual sentences as in our approach. Their choice is based on the assumption that texts on the Internet appear in complete paragraphs, with structure strictly adhere to the fixed training templates, which may be true for specific domains they test on, but fails to hold for domain-independent application. Instead, our algorithm aims to synthesize the article in the sentential level. We select sentences to fit the source content at run time, regardless to whether a pre-determined structural template exists or not. Therefore the requirement on the structures of source articles becomes very flexible, enabling our system to work for arbitrary domain. In a sense, rather than being a structure-aware approach, our algorithm performs in a content-aware manner. This also makes maintaining coherence throughout article a lot more challenging.

Works on monolingual extractive text summarization also lend insights into our problem. (Goldstein et al., 2000) used sequential sentence selection based on Maximal Marginal Relevance Multi-Document (MMR-MD) score to form summarizations for multiple documents, with the constraint of sentence count. Since our problem does not have this constraint, we employ a variant of MMR-MD and introduced new terms specific to this task. (Takamura and Okumura, 2009) formulated a text summarization task as a maximum coverage problem with knapsack constraint and proposed a variety of combinatorial mathematics-based algorithms for solving the optimization problem.

For multi-lingual summarization, (Evans, 2005) applied the concept of multi-lingual text similarity to summarization and improved readability of English summaries of Arabic text by replacing machine translated Arabic sentences with highly similar English sentences whenever possible.

3 Methodology

Figure 1 describes the high-level algorithm of our approach. The system takes as input the English Wikipedia page and outputs an article in Chinese.

First, the structured English article is extracted from the Wikipedia page. Due to the relative independence of contents in different sections in typical Wikipedia articles (e.g. childhood, early writings), a separate synthesis task is performed on each section and all synthesized sections are eventually combined in the original order to form the Chinese article.

For each section, keywords are extracted from the English text using both tf-idf and the graph-based TextRank algorithm. Named entities, time indicators, and terms with Wikipedia hyperlinks are also included. These keywords express the topics of the current section and are regarded as the content guideline. We then use Google Translate and Google Dictionary to

Input:

English version of an entry

Output:

Synthesized Chinese version

Algorithm:

1: Parse the English Wikipedia page to extract the structured texts.
2: For each section:
  2.1: Extract keywords.
  2.2: Use Chinese translation of keywords to search online Chinese texts.
  2.3: Filter retrieved Chinese texts and split them into segments.
  2.4: Synthesize the current section using candidate segments.
3: Generate the Chinese Wikipedia page by combining synthesized sections according to the original structure of English version.

Figure 1. High-level algorithm of the synthesis approach

obtain the Chinese translations of these keywords and thereby convert the content guideline into Chinese. The Chinese keywords are then combined with the translated subject term and section title to form queries that are used to retrieve online Chinese documents by Google search. The returned Chinese documents are clustered and filtered based on both their format and content. The remaining candidate excerpts are further split using the TextTiling algorithm (Hearst, 1997) into segments that constitutes the text units for synthesis. This unit size ensures both semantic completeness within each unit and flexibility of combining multiple units into coherent paragraphs. Segments are chosen according to scores computed iteratively by a variant of the MMR-MD scoring function that considers not only the relevance of an individual segment to the source section but also its impact on the provisional synthesized section as a whole.

3.1 Wikipedia Page Preprocessing

The source Wikipedia page is parsed to remove non-textual page elements (e.g. images, info-boxes and side-bars). Only texts and headings are extracted and their structures are maintained as templates for final integration of synthesized sections.

3.2 Keyword Extraction

The keyword set K for a section is the union of 6 categories of content-bearing terms.

$$K = \bigcup K_c$$

K_1: set of terms with high tf-idf score (top 5%)
K_2: set of terms with high TextRank score (top 5%)
K_3: set of named entities
K_4: set of temporal indicators (e.g. June, 1860)
K_5: set of terms with Wikipedia links
K_6: section title

For K_1, tf-idf scores are computed by:

$$tfidf_i = \sqrt{tf_i} \times \log\left(\frac{N}{df_i} + 1\right)$$

where tf_i is the term frequency of term *i* in the section and df_i is the document frequency of term *i* in a corpus consists of 2725 high-quality English Wikipedia articles [1], which well represent the language style of Wikipedia.

For K_2, we compute TextRank scores according to (Mihalcea and Tarau, 2004). It is a graph-based model where words as vertices recursively vote for the weights of their linked neighbors (e.g. words appear in the same sentence as them) using the formula:

$$WS(V_i) = (1-d) + d * \sum_{V_j \in In(V_i)} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} WS(V_j)$$

[1] http://evanjones.ca/software/wikipedia2text.html

Where $In(V_i)$ is the set of vertices with forward links to i, $Out(V_i)$ is the set of vertices receiving links from i, w_{ji} is the weight of edge between V_i and V_j. In the case of a word graph, we simplify this formula by assuming the graph to be undirected and unweighted. Each pair of words occurring in the same sentence share an edge between them and all word vertices have initial weights of 1.

Unlike tf-idf which considers only word-specific values and tends to give higher weights for rare words, TextRank uses global information about how a word is used in its context to induce its importance and has the advantage of highlighting keywords that are relatively common but highly relevant. In this sense, these two measures complement each other. Named entities are recognized using the named entity chunker provided by the NLTK (Natural Language ToolKit) package[2].

3.3 Keyword Translation

Keywords are then translated using Google Dictionary to form Chinese queries. Usually one English keyword has several translations and they will be used jointly when forming the search query.

Google Dictionary often fails to generate correct transliteration for rare names, so we augment it with a function of parenthesized phrase translation. We basically seeks named-entity strings from online documents that are in the format of '*CHINESE* (*ENGLISH*)' and extracts the Chinese transliteration from the pattern using regular expression combined with a Pinyin (Chinese Romanization) [3] /English pronunciation lookup table. Since Chinese words are not spaced in documents, the Pinyin/English lookup is helpful to determine the boundary of the Chinese transliteration based on the fact that most Chinese transliterations start with characters pronounced similar to the initial syllables in corresponding English names. This function is relatively simple but works surprisingly well as many rare named entities are available in this pattern on the Web.

3.4 Web Search

Keywords in Chinese alternatively form query pairs with the Wikipedia subject term. Each pair is used to retrieve a set of (16 in our experiments) Chinese documents containing both words with Google Search. If a keyword has multiple translations, they are joined by the string 'OR' in the query which is the way to specify alternatives in Google logic. If a keyword is a named entity, its English version is also used as an alternative in order to acquire documents in which the subject is referred to by its English name instead of transliterations. For the subject "Chekhov/契诃夫", a keyword with two transliterations "Taganrog/塔甘罗格/塔干罗格" and another keyword with two transliterations "father/父亲/爸爸" will result in two query pairs: "Chekhov OR 契诃夫 Taganrog OR 塔甘罗格 OR 塔干罗格" and "Chekhov OR 契诃夫 父亲 OR 爸爸".

3.5 Candidate Filtering

The retrieved excerpts are filtered first by criteria on format include text length and the percentage of white-space and non-Chinese characters. Pair-wise similarity is then computed among all the remaining excerpts and those above a certain threshold are clustered. Within a cluster only the centroid excerpt with maximum similarity with the source section will be selected. This stage typically eliminates ¾ of the documents that are either not sufficiently relevant or redundant. The similarity measure we use is a combination of both English and Chinese versions of cosine similarity and Jaccard index.

$$\mathrm{SIM}(\mathrm{a},\mathrm{b}) = 0.3 \times \mathrm{COS}_{\mathrm{EN}}(\mathrm{a},\mathrm{b}) + 0.3 \times \mathrm{COS}_{\mathrm{CH}}(\mathrm{a},\mathrm{b}) + 0.2 \times \mathrm{JAC}_{\mathrm{EN}}(\mathrm{a},\mathrm{b}) + 0.2 \times \mathrm{JAC}_{\mathrm{CH}}(\mathrm{a},\mathrm{b})$$

For Chinese excerpts, English similarity is computed by first translating them into English by Google Translate and taking tf-idf as token weights. Similar procedure works for computing Chinese similarity for English excerpts, except that Chinese texts need to be

[2] The package is available at http://www.nltk.org
[3] Pinyin information is obtained from Unicode Han Database at http://www.unicode.org/reports/tr38/

segmented[4] first and weights are based on tf only. These machine translations do not require grammatical correctness since they are essentially used as bags of words in both cosine similarity and Jaccard index. During this stage, every excerpt acquires bi-lingual versions, which is important for the extended similarity measure in the iterative ranking function.

Filtered excerpts are further split into segments using the TextTiling algorithm. After clustering the remaining segments form the candidate units for synthesis of the current section.

3.6 Iterative Scoring Function

Based on the idea that the ‘goodness’ of a segment should be evaluated both on its individual relevance to the source and the overall impact on the synthesized section, we summarize four factors for scoring a segment: (1) Intuitively a segment scores higher if it has higher similarity to the source section; (2) A segment makes positive contribution to synthesized section if it introduces some keywords mentioned in the source; (3) A segment tends to improve the coherence of synthesized section if it comes from the same excerpts as the other segments in synthesized section; (4) A sentence should be penalized if its content is redundant with the synthesized section.

Integrating the four factors above, we propose that for source text *r*, the score of the *i*th candidate segment s_i in the *n*th iteration is formulated as:

$$Q_{r,n}(s_i) = w_s \times S_r(s_i) + w_k \times K_{r,n}(s_i) + w_c \times C_n(s_i) - w_R \times R_n(s_i)$$

This formula is composed of 4 terms corresponding to the ‘goodness’ factors: $S_r(s_i)$ for similarity, $K_{r,n}(s_i)$ for keyword coverage, $C_n(s_i)$ for coherence, and $R_n(s_i)$ for redundancy. The corresponding weights are tuned in a large number of experiments as to achieve optimal performance. This function is a variant of the original MMR-MD score tailored for our application.

$S_r(s_i)$ is a comprehensive similarity measure of segment s_i to the reference text r.

$$S_r(s_i) = w_1 \times SIM(s_i, \mathrm{r}) + w_2 \times SIM(s_i, \mathrm{p}) + w_3 \times SIM(e_i, \mathrm{r}) + w_4 \times SIM(e_i, \mathrm{p})$$

where p is the parent section of r and e_i is the parent excerpt of s_i. Similarities between parent excerpts are also examined because sometimes two segments, especially short segments, despite their textual similarity actually come from very different contexts and exhibit different focuses. In this case, the latter three terms will suppress the score between these two segments which would otherwise be erroneously high and therefore produce a more precise measure of similarity.

$\mathrm{K}_{\mathrm{r,n}}(\mathrm{s_i})$ measures the contribution of $\mathrm{s_i}$ in terms of uncovered keywords.

$$\mathrm{K}_{\mathrm{r,n}}(\mathrm{s_i}) = \sum_{\substack{\mathrm{k} \in \mathrm{U}_{\mathrm{r,n}} \\ \mathrm{k} \neq \mathrm{subject}}} \mathrm{idf}(\mathrm{k})$$

$$\mathrm{U}_{\mathrm{r,n}} = \mathrm{K_r} - \bigcup_{\mathrm{s_j} \in \mathrm{D_n}} \mathrm{K_j}$$

where $\mathrm{D_n}$ is the winner set in the nth iteration. $\mathrm{K_r}$ is the set of keywords in the reference text and $\mathrm{K_j}$ is the set of keywords in the selected segment $\mathrm{s_j}$. $\mathrm{U}_{\mathrm{r,n}}$ represents the set of keywords in the reference that are not yet been covered by the provisional synthesized text in the nth iteration. $\mathrm{K}_{\mathrm{r,n}}(\mathrm{s_i})$ quantifies the keyword contribution as the sum of idf values of uncovered keywords. The subject term is excluded because it as a keyword does not reflect any topic bias and is therefore not a good indicator for coverage.

$C_n(s_i)$ is a term that reflects the coherence and readability in the synthesized text.

$$C_n(s_i) = \left|\{\mathrm{s_j} | \mathrm{e_j} = \mathrm{e_i}, \mathrm{s_j} \in D_n\}\right|$$

[4] The segmentation tool using forward maximum matching is obtained at http://technology.chtsai.org/mmseg

where e_i is the parent excerpt of s_i and e_j is the parent excerpt of s_j. Segments from the same excerpts tend to be less redundant and more coherent. Therefore candidates that share the same parent excerpts as segments in winner set are more favorable and rewarded by this term. This is a major difference from the original MMR-MD function in which sentences from different documents are favored. This is because their formula is targeted for automatic summarization where more emphasis is put on diversity rather than coherence.

$R_n(s_i)$ measures the redundancy of the synthesized text if s_i is included. It is quantified as the maximum similarity of s_i with all selected segments.

$$R_n(s_i) = \max_{s_j \in D_n} S(s_i, s_j)$$

3.7 Segment Selection Algorithm

Figure 2 describes the segment selection algorithm. Starting with a candidate set and an empty winner set, we iteratively rank the candidates by Q and in each iteration the top-ranked segment is examined. There are two circumstances a segment would be selected for the winner set:

(1) if the segment scores sufficiently high
(2) the segment does not score high enough for an unconditional selection, but as long as it introduces uncovered keywords, its contribution to the overall content quality may still overweigh the compromised similarity

In the second circumstance however, since we are only interested in the uncovered keywords, it may not be necessary for the entire segment to be included in the synthesized text. Instead, we only include the sentences in this segment that contain those keywords. Therefore we propose two conditions:

- ***C**~sel-segment~*: condition for selecting a segment
$$Q_{r,n}(s_{top}) > 0.8 * \mathrm{Q}_{\max}$$

- ***C**~sel-sentence~*: condition for selecting sentences
$Q_{r,n}(\mathrm{s_{stop}}) > 0.6 * \mathrm{Q}_{\max}$ **and** $K_{r,n}(\mathrm{s_{stop}}) > 0$ **and** $S_r(s_{top}) > 0.3 * \mathrm{S}_{\max}$

Thresholds in both conditions are not static but dependent on the highest score of all candidates in order to accommodate diversity in score range for different texts. Finally if no more candidates are able to meet the lowered score threshold, even if they might carry new keywords, we assume they are not suitable for synthesis and return the current winner set. This break condition is formulated as $\boldsymbol{C_{break}}$:

- $\boldsymbol{C_{break}}$: condition to finish selection
$$Q_{r,n}(s_{top}) < 0.6 * \mathrm{Q}_{\max}$$

Input:
  $\boldsymbol{S_n}$: candidate set in iteration *n*
  $\boldsymbol{r}$: the reference text
Define:
  $\boldsymbol{n}$: iteration index
  $\boldsymbol{D_n}$: winner set in iteration *n*
  $\boldsymbol{C_{sel\text{-}segment}}$: $Q_{r,n}(s_{top}) > 0.8 * \mathrm{Q}_{\max}$
  $\boldsymbol{C_{sel\text{-}sentence}}$: $Q_{r,n}(s_{top}) > 0.6 * \mathrm{Q}_{\max}$
    and $K_{r,n}(s_{top}) > 0$
    and $S_r(s_{top}) > 0.3 * \mathrm{S}_{\max}$
  $\boldsymbol{C_{break}}$: $Q_{r,n}(s_{top}) < 0.6 * \mathrm{Q}_{\max}$
Algorithm:
  $\boldsymbol{D_n} \leftarrow \emptyset, \boldsymbol{n} \leftarrow 0$
  while $\boldsymbol{S_n} \neq \emptyset$:
    $\boldsymbol{s_{top}} \leftarrow \arg\max_{\mathbf{s_i \in S_n}} \boldsymbol{Q_{r,n}(s_i)}$
    if $\mathbf{C_{break}}$:
      return $\boldsymbol{D_n}$
    else if $\mathbf{C_{sel\text{-}segment}}$:
      $\boldsymbol{D_n} \leftarrow \boldsymbol{D_n} + \boldsymbol{s_{top}}$
    else if $\mathbf{C_{sel\text{-}sentence}}$:
      $\boldsymbol{D_n} \leftarrow \boldsymbol{D_n} +$ sentences in $\boldsymbol{s_{top}}$ with the uncovered keywords
      $\boldsymbol{S_n} \leftarrow \boldsymbol{S_n} - \boldsymbol{s_{top}}$
      $\boldsymbol{n} \leftarrow \boldsymbol{n} + \mathbf{1}$
Output:
  Synthesized text for the reference ***r***

Figure 2. Segment selection algorithm

4 Evaluation

4.1 Experiment Setup

We evaluate our system on 16 Wikipedia subjects across 5 different domains as listed in Table 1.

Category	Subjects
Person	Anton Chekhov Abu Nuwas Joseph Haydn Li Bai
Organization	HKUST IMF WTO
Events	Woodstock Festival Invasion of Normandy Decembrist Revolt
Science	El Nino Gamma Ray Stingray
Culture	Ceramic Art Spiderman Terrorism

Table 1. Subjects used for evaluation

The subjects are selected from "the List of Articles Every Wikipedia Should Have" [5] published by Wikimedia. These subjects are especially appropriate for our evaluation because we can (1) use a subset of such articles that have high quality in both English and Chinese as standard reference for evaluation; (2) safely assume Chinese information about these subjects is widely available on the Internet; (3) take subjects currently without satisfactory versions in Chinese as our challenge.

Human Evaluation

We presented the synthesized articles of these subjects to 5 native Chinese readers who compare synthesized articles with MT results and existing Chinese versions on Wikipedia which range from translated stubs to human-authored segments. We asked the reviewers to score them on a 5-point scale in terms of four quality indicators: structural similarity to the English version, keyword coverage, fluency, and conciseness.

Automatic Evaluation

In addition to human evaluation, we also compare synthesized articles to several high-quality Chinese Wikipedia articles using ROUGE-L (C.Y. Lin, 2004). We assume these Chinese versions are the goals for our synthesis system and greater resemblance with these standard references indicates better synthesis. ROUGE-L measures the longest common subsequence (LCS) similarity between two documents, rather than simply word overlap so it to some degree reflects fluency.

[5] http://meta.wikimedia.org/wiki/List_of_articles_every_Wikipedia_should_have/Version_1.2

4.2 Result Analysis

Human Evaluation

Human evaluator feedbacks for articles in different categories are shown in Table 2. Machine-translated versions are judged to have the highest score for structural similarity, but erroneous grammar and word choices make their readability so poor even within sentences and therefore of no practical use.

Generally, articles synthesized by our system outperform most existing Chinese versions in terms of both structural and content similarity. Many existing Chinese versions completely ignore important sections that appear in English versions, while our system tries to offer information with as much fidelity to the English version as possible and is usually able to produce information for every section. Synthesized articles however, tend to be less fluent and more redundant than human-authored versions.

Performance varies in different domains. Synthesis works better for subjects in *Person* category, because the biographical structure provides a specific and fairly unrelated content in each section, making the synthesis less redundancy-prone. On the other hand, there is arbitrariness when organizing articles in *Event* and *Culture* category. This makes it difficult to find online text organized in the same way as the English Wikipedia version, therefore introducing a greater challenge in sentence selection for each section. Articles in the *Science* category usually include rare terminologies, and formatted texts like diagrams and formula, which impede correct translation and successful extraction of keywords.

Cat.	Structural Similarity			Coverage			Fluency			Conciseness		
	Synt.	Orig.	MT	**Synt.**	Orig.	MT	**Synt.**	Orig.	MT	**Synt.**	Orig.	MT
Psn.	**2.85**	1.49	5	**2.94**	1.84	4.51	**2.71**	4.58	0.83	**1.74**	4.47	n/a
Org.	**1.96**	1.22	5	**2.51**	2.10	4.46	**2.10**	4.42	1.06	**0.99**	4.53	n/a
Evt.	**1.37**	1.13	5	**2.56**	1.94	4.40	**2.45**	4.46	0.81	**0.80**	4.40	n/a
Sci.	**2.43**	1.30	5	**2.68**	2.14	4.42	**2.53**	4.51	1.02	**1.05**	4.50	n/a
Cul.	**1.39**	1.35	5	**2.2**	2.21	4.54	**2.32**	4.54	0.94	**1.34**	4.59	n/a
Avg.	**2.02**	1.30	5	**2.58**	2.05	4.47	**2.42**	4.50	0.93	**1.22**	4.50	n/a

Table 2. Result of human evaluation against English source articles (out of 5 points; Synt: synthesized articles; Orig: the existing human-authored Chinese Wikipedia versions; MT: Chinese versions generated by Google Translate)

Automatic Evaluation

Using ROUGE-L to measure the quality of both synthesized and MT articles against human-authored standard references, we find synthesized articles generally score higher than MT versions. The results are shown in Table 3.

Category	Recall		Precision		F-score	
	Synt.	MT	Synt.	MT	Synt.	MT
Psn.	**0.48**	0.30	**0.20**	0.16	**0.28**	0.22
Org.	**0.40**	0.29	**0.16**	0.13	**0.23**	0.18
Evt.	**0.36**	0.26	0.13	**0.15**	**0.19**	0.19
Sci.	**0.31**	0.22	**0.14**	0.11	**0.19**	0.15
Cul.	**0.37**	0.27	**0.13**	0.12	**0.24**	0.17
Avg.	**0.38**	0.27	**0.15**	0.13	**0.23**	0.18

Table 3. Results of automatic evaluation against gold Chinese reference articles (Synt: synthesized articles; MT: Chinese versions generated by Google Translate)

The synthesized articles, extracted from high quality human-authored monolingual texts, are generally better in precision than the MT articles because there is less erroneous word choice or grammatical mistakes. Most synthesized articles also have higher recall than MT versions because usually a substantial portion of the high-quality Chinese excerpts, after being retrieved by search engine, will be judged by our system as good candidate texts and included into the synthesized article. This naturally increases the resemblance of synthesized articles to standard references, and thus the F-scores. Note that since our method is unsupervised, the inclusion of the standard Chinese articles underscores the precision and recall of our method.

5 Conclusion

In this paper, we proposed an unsupervised approach of synthesizing Wikipedia articles in multiple languages based on an existing high-quality version of any entry. By extracting keywords from the source article and retrieving relevant texts from the monolingual Web in a target language, we generate new articles using an iterative scoring function.

Synthesis results for several subjects across various domains confirmed that our method is able to produce satisfactory articles with high resemblance to the source English article. For many of the testing subjects that are in ‘stub’ status, our synthesized articles can act as either replacement or supplement to existing Chinese versions. For other relatively well-written ones, our system can help provide content prototypes for missing sections and missing topics, bootstrapping later human editing.

A weakness of our system is the insufficient control over coherence and fluency in paragraph synthesis *within* each section, new methods are being developed to determine the proper order of chosen segments and optimize the readability.

We are working to extend our work to a system that supports conversion between major languages such as German, French and Spanish. The employment of mostly statistical methods in our approach facilitates the extension. We have also released a downloadable desktop application and a web application based on this system to assist Wikipedia users.

Reference

Adafre, Sisay F. and Maarten de Rijke, “Finding Similar Sentences across Multiple Languages in Wikipedia”, *Proceedings of the EACL Workshop on New Text*, Trento, Italy, 2006

Bird, Steven, E. Klein, and E. Loper, *Natural Language Processing with Python --- Analyzing Text with the Natural Language Toolkit*, O'Reilly Media, 2009

Evans, David K., “Identifying Similarity in Text: Multi-Lingual Analysis for Summarization”, PhD thesis, Columbia University, 2005.

Goldstein, Jade, Vibhu Mittal, Jaime Carbonell and Mark Kantrowitz, “Multi-document summarization by sentence extraction”, *NAACL-ANLP 2000 Workshop on Automatic summarization*, pages 40-48, 2000

Hearst, Marti A., “TextTiling: Segmenting Text into Multi-paragraph Subtopic Passages”, *Computational Linguistics*, Volume 23, Issue 1, pp. 33-64, 1997

Lin, Chin-Yew, “ROUGE: A Package for Automatic Evaluation of Summaries”, *Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004*, Barcelona, Spain.

Mihalcea, Rada and Paul Tarau, “TextRank: Bringing order into texts”, *Proceedings of EMNLP*, pages 404–411 Barcelona, Spain, 2004

Sauper, Christina and Regina Barzilay, “Automatically Generating Wikipedia Articles: a Structure-Aware Approach”, *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP*, pages 208–216, Suntec, Singapore, 2-7 August 2009.

Takamura, Hiroya and Manabu Okumura, “Text Summarization Model based on Maximum Coverage Problem and its Variant”, *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 781-789, 2009

Simplicity is Better: Revisiting Single Kernel PPI Extraction

Sung-Pil Choi
Information Technology Laboratory
Korea Institute of Science and Technology Information
spchoi@kisti.re.kr

Sung-Hyon Myaeng
Department of Computer Science
Korea Advanced Institute of Science and Technology
myaeng@kaist.ac.kr

Abstract

It has been known that a combination of multiple kernels and addition of various resources are the best options for improving effectiveness of kernel-based PPI extraction methods. These supplements, however, involve extensive kernel adaptation and feature selection processes, which attenuate the original benefits of the kernel methods. This paper shows that we are able to achieve the best performance among the state-of-the-art methods by using only a single kernel, convolution parse tree kernel. In-depth analyses of the kernel reveal that the keys to the improvement are the tree pruning method and consideration of tree kernel decay factors. It is noteworthy that we obtained the performance without having to use any additional features, kernels or corpora.

1 Introduction

Protein-Protein Interaction (PPI) Extraction refers to an automatic extraction of the interactions between multiple protein names from natural language sentences using linguistic features such as lexical clues and syntactic structures. A sentence may contain multiple protein names and relations, i.e., multiple PPIs. For example, the sentence in Fig.1 contains a total of six protein names of varying word lengths and three explicit interactions (relations). The interaction type between *phosphoprotein* and the acronym *P* in the parentheses is "*EQUAL*." A longer protein name *phosphoprotein of vesicular stomatitis virus* is related to *nucleocapsid protein* via "*INTERACT*" relation. Like the first PPI, *nucleocapsid protein* is equivalent to the abbreviated term *N*.

It is not straightforward to extract PPIs from a sentence or textual segment. There may be multiple protein names and their relationships, which are intertwined in a sentence. An interaction type may be expressed in a number of different ways.

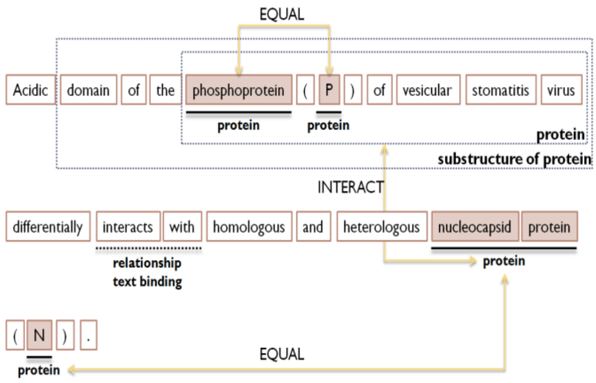


Figure 1. An example sentence containing multiple PPIs involving different names of varying scopes and relations[1]

A significant amount of efforts have been devoted to kernel-based approaches to PPI extractions (PPIE) as well as relation extractions[2] (Zhang et al., 2006; Pyysalo et al., 2008; GuoDong et al., 2007; Zhang et al., 2008; Airola et al., 2008; Miwa et al., 2009). They include word feature kernels, parse tree kernels, and graph kernels. One of the benefits of using a kernel method is that it can keep the original

[1] BioInfer, Sentence ID:BioInfer.d10.s0

[2] Relation extraction has been studied massively with the help of the ACE (www.nist.gov/tac) competition workshop and its corpora. The ACE corpora contain valuable information showing the traits of target entities (e.g., entity types, roles) for relation extraction in single sentences. Since all target entities are of the same type, protein name, in PPIE, however, we cannot use relational information that exists among entity types. This makes PPIE more challenging.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 206–214, Beijing, August 2010

formation of target objects such as parse trees, not requiring extensive feature engineering for learning algorithms (Zelenko et al., 2003).

In an effort to improve the performance of PPIE, researchers have developed not only new kernels but also methods for combining them (GuoDong et al., 2007; Zhang et al., 2008; Airola et al., 2008; Miwa et al., 2009a; Miwa et al., 2009b). While the intricate ways of combing various kernels and using extra resources have played the role of establishing strong baseline performance for PPIE, however, they are viewed as another form of engineering efforts. After all, one of the reasons the kernel methods have become popular is to avoid such engineering efforts.

Instead, we focus on a state-of-the-art kernel and investigate how it can be best utilized for enhanced performance. We show that even with a single kernel, convolution parse tree kernel in this case, we can achieve superior performance in PPIE by devising an appropriate preprocessing and factor adjustment method. The keys to the improvement are tree pruning and consideration of a tree kernel decay factor, which are independent of the machine learning model used in this paper. The main contribution of our work is the extension and application of the particular convolution tree kernel method for PPIE, which gives a lesson that a deep analysis and a subsequent extension of a kernel for maximal performance can override the gains obtained from engineering additional features or combining other kernels.

The remaining part of the paper is organized as follows. In section 2, we survey the existing approaches. Section 3 introduces the parse tree kernel model and its algorithm. Section 4 explains the performance improving factors applied to the parse tree kernel. The architecture of our system is introduced in section 5. Section 6 shows the improvements in effectiveness in multiple PPI corpora and finally we conclude our work in section 7.

2 Related Work

In recent years, numerous studies have attempted to extract PPI automatically from text. Zhou and He (2008) classified various PPIE approaches into three categories: linguistic, rule-based and machine learning and statistical methods.

Linguistic approaches involve constructing special grammars capable of syntactically expressing the interactions in sentences and then applying them to the language analyzers such as part-of-speech taggers, chunkers and parsers to extract PPIs. Based on the level of linguistic analyses, we can divide the linguistic approaches into two categories: shallow parsing (Sekimizu et al., 1998; Gondy et al., 2003) and full parsing methods (Temkin & Gilder, 2003; Nikolai et al., 2004).

Rule-based approaches use manually defined sets of lexical patterns and find text segments that match the patterns. Blaschke et al. (1996) built a set of lexical rules based on clue words denoting interactions. Ono et al. (2001) defined a group of lexical and syntactic interaction patterns, embracing negative expressions, and applied them to extract PPIs from documents about "*Saccharomyces cerevisiae*" and "*Escherichia coli*". Recently, Fundel et al. (2007) proposed a PPI extraction model based on more systematic rules using a dependency parser.

Machine learning and statistical approaches have been around for a while but have recently become a dominant approach for PPI extraction. These methods involve building supervised or semi-supervised models based on training sets and various feature extraction methods (Andrade & Valencia, 1998; Marcotte et al., 2001; Craven & Kumlien, 1999). Among them, kernel-based methods have been studied extensively in recent years. Airola et al. (2008) attempted to extract PPIs using a graph kernel by converting dependency parse trees into the corresponding dependency graphs.

Miwa et al. (2009a) utilized multiple kernels such as word feature kernels, parse tree kernels, and even graph kernels in order to improve the performance of PPI extraction. Their experiments based on five PPI corpora, however, showed that combining multiple kernels gave only minor improvements compared to other methods. To further improve the performance of the multiple kernel system, the same group combined multiple corpora to exploit additional features for a modified SVM model (Miwa et al., 2009b). While they achieved the best performance in PPI extraction, it was possible only with additional kernels and corpora from which additional features were extracted.

Unlike the aforementioned approaches trying to use all possible resources for performance enhancement, this paper aims at maximizing the performance of PPIE using only a single kernel without any additional resources. Without lowering the performance, we attempt to stick to the initial benefits of the kernel methods: *simplicity* and *modularity* (Shawe-Taylor & Cristianini, 2004).

3 Convolution Parse Tree Kernel Model for PPIE

The main idea of a convolution parse tree kernel is to sever a parse tree into its sub-trees and transfer it as a point in a vector space in which each axis denotes a particular sub-tree in the entire set of parse trees. If this set contains M unique sub-trees, the vector space becomes M-dimensional. The similarity between two parse trees can be obtained by computing the inner product of the two corresponding vectors, which is the output of the parse tree kernel.

There are two types of parse tree kernels of different forms of sub-trees: one is *SubTree Kernel* (*STK*) proposed by Vishwanathan and Smola (2003), and the other is *SubSet Tree Kernel* (*SSTK*) developed by Collins and Duffy (2001). In *STK*, each sub-tree should be a complete tree rooted by a specific node in the entire tree and ended with leaf nodes. All the sub-trees must obey the production rules of the syntactic grammar. Meanwhile, *SSTK* can have any forms of sub-trees in the entire parse tree given that they should obey the production rules. It was shown that *SSTK* is much superior to *STK* in many tasks (Moschitti, 2006). He also introduced a fast algorithm for computing a parse tree kernel and showed its beneficial effects on the semantic role labeling problem.

A parse tree kernel can be computed by the following equation:

$$K(T_1,T_2,\lambda,\sigma) = \sum_{n_1\in N_{T_1}} \sum_{n_2\in N_{T_2}} \Delta(n_1,n_2,\lambda,\sigma) \quad (1)$$

where T_i is i^{th} parse tree and n_1 and n_2 are nodes in N_T, the set of the entire nodes of T. λ represents a tree kernel decay factor, which will be explained later, and σ decides the way the tree is severed. Finally $\Delta(n_1, n_2, \lambda, \sigma)$ counts the number of the common sub-trees of the two parse trees rooted by n_1 and n_2. Figure 2 shows the algorithm.

In this algorithm, the *get_children_number* function returns the number of the direct child nodes of the current node in a tree. The function named *get_node_value* gives the value of a node such as part-of-speeches, phrase tags and words. The *get_production_rule* function finds the grammatical rule of the current node and its children by inspecting their relationship.

```
FUNCTION delta(TreeNode n1, TreeNode n2, λ, σ)
n1 = one node of T1;  n2 = one node of T2;
λ = tree kernel decay factor;  σ = tree division me-
thod;
BEGIN
   nc1 = get_children_number(n1);
   nc2 = get_children_number(n2);
   IF nc1 EQUAL 0 AND nc2 EQUAL 0 THEN
      nv1 = get_node_value(n1);
      nv2 = get_node_value(n2);
      IF nv1 EQUAL nv2 THEN RETURN 1;
   ENDIF
   np1 = get_production_rule(n1);
   np2 = get_production_rule(n2);
   IF np1 NOT EQUAL np2 THEN RETURN 0;

   IF np1 EQUAL np2 AND nc1 EQUAL 1
           AND nc2 EQUAL 1 THEN
      RETURN λ;
   END IF

   mult_delta = 1;
   FOR I = 1 TO nc1
      nch1 = I^th child of n1;   nch2 = I^th child of n2;
      mult_delta = mult_delta ×
                     (σ + delta(nch1, nch2, λ, σ));
   END FOR
   RETURN λ × mult_delta;
END
```

Figure 2. Δ (n_1, n_2, λ, σ) algorithm

4 Performance Improving Factors

4.1 Tree Pruning Methods

Tree pruning for relation extraction was firstly introduced by Zhang et al. (2006) and also referred to as "*tree shrinking task*" for removing less related contexts. They suggested five types of the pruning methods and later invented two more in Zhang et al. (2008). Among them, the path-enclosed tree (*PT*) method was shown to give the best result in the relation extraction task based on ACE corpus. We opted for this pruning method in our work.

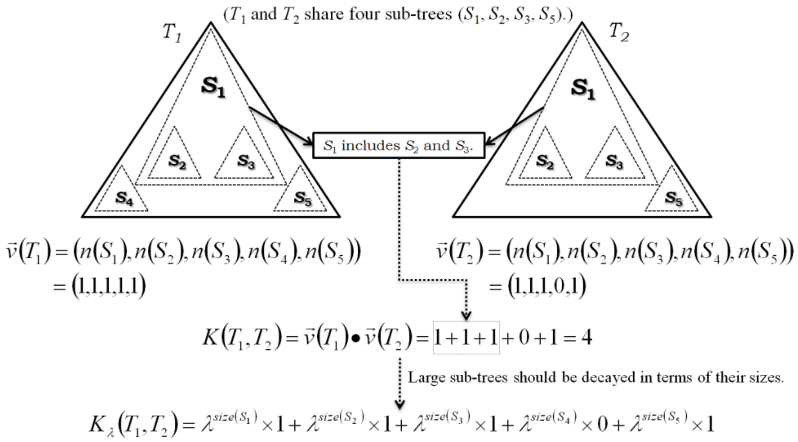


Figure 4. The effect of decaying in comparing two trees. $n(\cdot)$ denotes #unique subtrees in a tree.

Figure 3 shows how the *PT* method prunes a tree. To focus on the pivotal context, it preserves only the syntactic structure encompassing the two proteins at hand and the words in between them (the part enclosed by the dotted lines). Without pruning, all the words like *addition*, *increased* and *activity* would intricately participate in deciding the interaction type of this sentence.

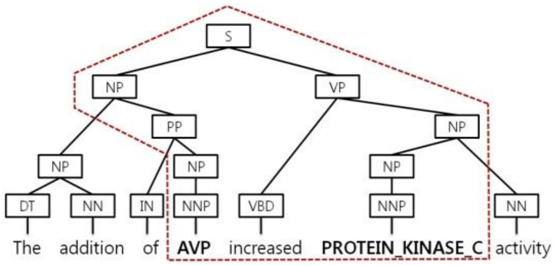


Figure 3. Path-enclosed Tree (PT) Method

Another important effect of the tree pruning is its ability to separate features when two or more interactions exist in a sentence. As in Figure 1, each interaction involves its unique context even though a sentence has multiple interactions. With tree pruning, it is likely to extract context-sensitive features by ignoring external features.

4.2 Tree Kernel Decay Factor

Collins and Duffy (2001) addressed two problems of the parse tree kernel. The first one is that its kernel value tends to be largely dominated by the size of two input trees. If they are large in size, it is highly probable for the kernel to accumulate a large number of overlapping counts in computing their similarity. Secondly, the kernel value of two identical parse trees can become overly large while the value of two different parse trees is much tiny in general. These two aspects can cause a trouble during a training phase because pairs of large parse trees that are similar to each other are disproportionately dominant. Consequently, the resulting models could act like nearest neighbor models (Collins and Duffy, 2001).

To alleviate the problems, Collins and Duffy (2001) introduced a scalability parameter called decay factor, $0 < \lambda \leq 1$ which scales the relative importance of tree fragments with their sizes as in line 33 of Fig. 2. Based on the algorithm, a decay factor decreases the degree of contribution of a large sub-tree exponentially in kernel computation. Figure 4 illustrates both the way a tree kernel is computed and the effect of a decay factor. In the figure, T_1 and T_2 share four common sub-trees (S_1, S_2, S_3, S_5). Let us assume that there are only two trees in a training set and only five unique sub-trees exist. Then each tree can be expressed by a vector whose elements are the number of particular sub-trees. Kernel value is obtained by computing the inner product of the two vectors. As shown in the figure, S_1 is a large sub-sub-trees, S_1, S_2 S_3, and S_4, two of which (S_2, and S_3) are duplicated in the inner product computation. It is highly probable for large sub-trees to contain many smaller sub-trees, which lead to an over-estimated similarity value between two parse trees. As mentioned above, therefore, it is necessary to rein those large sub-trees with respect to their sizes in computing kernel values by using decay factors. In this paper, we treat the decay factor as one of the important optimization parameters for a PPI extraction task.

5 Experimental Results

In order to show the superiority of the simple kernel based method using the two factors used in this paper, compared to the resent results for PPIE using additional resources, we ran a series of experiments using the same PPI corpora cited in the literature. In addition, we show that the method is robust especially for cross-corpus experiments where a classifier is trained and tested with entirely different corpora.

5.1 Evaluation Corpora

To evaluate our approach for PPIE, we used "*Five PPI Corpora*[3]" organized by Pyysalo et al. (2008). It contains five different PPI corpora: AImed, BioInfer, HPRD50, IEPA and LLL. They have been combined in a unified XML format and "*binarized*" in case of involving multiple interaction types.

	AIMed	BioInfer	HPRD50	IEPA	LLL
#Sentence	1,955	1,100	145	486	77
#Positive	1,000	2,534	163	335	164
#Negative	4,834	7,132	270	482	166

Table 1. Five PPI Corpora

Table 1 shows the size of each corpus in "*Five PPI Corpora*." As mentioned before, a sentence can have multiple interactions, which results in the gaps between the number of sentences and the sum of the number of instances. Negative instances have been automatically generated by enumerating sentences with multiple proteins but not having interactions between them (Pyysalo et al., 2008).

5.2 Evaluation Settings

In order to parse each sentence, we used Charniak Parser[4]. For kernel-based learning, we expanded the original *libsvm* 2.89[5] (Chang & Lin, 2001) so that it has two additional kernels including parse tree kernel and composite kernel[6] along with four built-in kernels[7]

Our experiment uses both macro-averaged and micro-averaged *F*-scores. Macro-averaging computes *F*-scores for all the classes individually and takes average of the scores. On the other hand, micro-averaging enumerates both positive results and negative results on the whole without considering the score of each class and computes total *F*-score.

In 10-fold cross validation, we apply the same split used in Airola et al., (2008), Miwa et al., (2009a) and Miwa et al., (2009b) for comparisons. Also, we empirically estimate the regularization parameters of SVM (*C*-values) by conducting 10-fold cross validation on each training data. We do not adjust the SVM thresholds to the optimal value as in Airola et al., (2008) and Miwa et al., (2009a).

5.3 PPI Extraction Performance

Table 2 shows the best scores of our system. The optimal decay factor varies with each corpus. In LLL, the optimal decay factor is 0.2[8] indicating that the shortage of data has forced our system to normalize parse trees more intensively with a strong decay factor in kernel computation in order to cover various syntactic structures.

	DF	*AC*	*ma*-P	*ma*-R	*ma*-F	$\sigma_{ma\text{-F}}$
A	0.6	83.6	72.8 (55.0)	62.1 (68.8)	**67.0** (60.8)	4.5 (6.6)
B	0.5	79.8	74.5 (65.7)	70.9 (71.1)	**72.6** (68.1)	2.7 (3.2)
H	0.7	74.5	75.3 (68.5)	71.0 (76.1)	**73.1** (70.9)	10.2 (10.3)
I	0.6	74.2	74.1 (67.5)	72.2 (78.6)	**73.1** (71.7)	6.0 (7.8)
L	0.2	82.2	83.2 (77.6)	81.2 (86.0)	**82.1** (80.1)	10.4 (14.1)

Table 2. The highest results of the proposed system w.r.t. decay factors. DF: Decay Factor, *AC*: accuracy, *ma*-F: macro-averaged F1, $\sigma_{ma\text{-F}}$: standard deviation of F-scores in CV. A:AIMed, B:BioInfer, H:HPRD50, I:IEPA, L:LLL. The numbers in parentheses refer to the scores of Miwa et al., (2009a).

Our system outperforms the previous results as in Table 2. Even using rich feature vectors including Bag-Of-Words and shortest path trees

[3] http://mars.cs.utu.fi/PPICorpora/eval-standard.html
[4] http://www.cs.brown.edu/people/ec/#software
[5] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[6] A kernel combining built-in kernels and parse tree kernel
[7] Linear, polynomial, radial basis function, sigmoid kernels

[8] It was determined by increasing it by 0.1 progressively through 10-fold cross validation.

generated from multiple corpora, Miwa et al., (2009b) reported 64.0% and 66.7% in AIMed and BioInfer, respectively. Our system, however, produced 67.0% in AIMed and 72.6% in BioInfer with a single parse tree kernel. We did not have to perform any intensive feature generation tasks using various linguistic analyzers and more importantly, did not use any additional corpora for training as done in Miwa et al., (2009b). While the performance differences are not very big, we argue that obtaining higher performance values is significant because the proposed system did not use any of the additional efforts and resources.

To investigate the effect of the scaling parameter of the parse tree kernel in PPI extraction, we measure how the performance changes as the decay factor varies (Figure 5). It is obvious that the decay factor influences the overall performance of PPI extraction. Especially, the *F*-scores of the small-scale corpora such as HPRD50 and LLL are influenced by the decay factor. The gaps between the best and worst scores in LLL and HPRD50 are 19.1% and 5.2%, respectively. The fluctuation in *F*-scores of the large-scale corpora (AIMed, BioInfer, IEPA) is not so extreme, which seems to stem from the abundance in syntactic and lexical forms that reduce the normalizing effect of the decay factor. The increase in the decay factor leads to the increase in the precision values of all the corpora except for LLL. The phenomenon is fairly plausible because the decreased normalization power causes the system to compute the tree similarities more intensively and therefore it classifies each instance in a strict and detailed manner. On the contrary, the recall values slightly decrease with respect to the decay factor, which indicates that the tree pruning (*PT*) has already conducted the normalization process to reduce the sparseness problem in each corpus.

Most importantly, along with tree pruning, decay factor could boost the performance of our system by controlling the rigidness of the parse tree kernel in PPI extraction.

Table 3 shows the results of the cross-corpus evaluation to measure the generalization power of our system as conducted in Airola et al., (2008) and Miwa et al., (2009a). Miwa et al., (2009b) executed a set of combinatorial experiments by mixing multiple corpora and presented their results. Therefore, it is not reasonable to compare our results with them due to the size discrepancy between training corpora. Nevertheless, we will compare our results with their approaches in later based on AIMed corpus.

As seen in Table 3, our system outperforms the existing approaches in almost all pairs of corpora. In particular, in the multiple corpora-based evaluations aimed at AIMed which has been frequently used as a standard set in PPI extraction, our approach shows prominent results compared with others. While other approaches showed the performance ranging from 33.3% to 60.8%, our approach achieved much higher scores between 55.9% and 67.0%. More specific observations are:

(1) Our PPIE method trained on any corpus except for IEPA outperforms the other approaches regardless of the test corpus only with a few exceptions with IEPA and LLL.

(2) Even when using LLL or HPRD50, two smallest corpora, as training sets, our system performs well with every other corpus for testing. It indicates that our approach is much less vulnerable to the sizes of training corpora than other methods.

(3) The degree of score fluctuation of our system across different testing corpora is much smaller than other regardless of the training data set. When trained on LLL, for example, the range for our system (55.9% ~ 82.1%) is smaller than the others (38.6% ~ 83.2% and 33.3% ~ 76.8%).

(4) The cross-corpus evaluation reveals that our method outperforms the others significantly. This is more visibly shown especially when the large-scale corpora (AIMed and BioInfer) are used.

(5) PPI extraction model trained on AIMed shows lower scores in IEPA and LLL as compared with other methods, which could trigger further investigation.

In order to convince ourselves further the superiority of the proposed method, we compare it with other previously reported approaches. Table 4 lists the macro-averaged precision, recall and *F*-scores of the nine approaches tested on AIMed. While the experimental settings are different as reported in the literature, they are quite close in terms of the numbers of positive and negative documents.

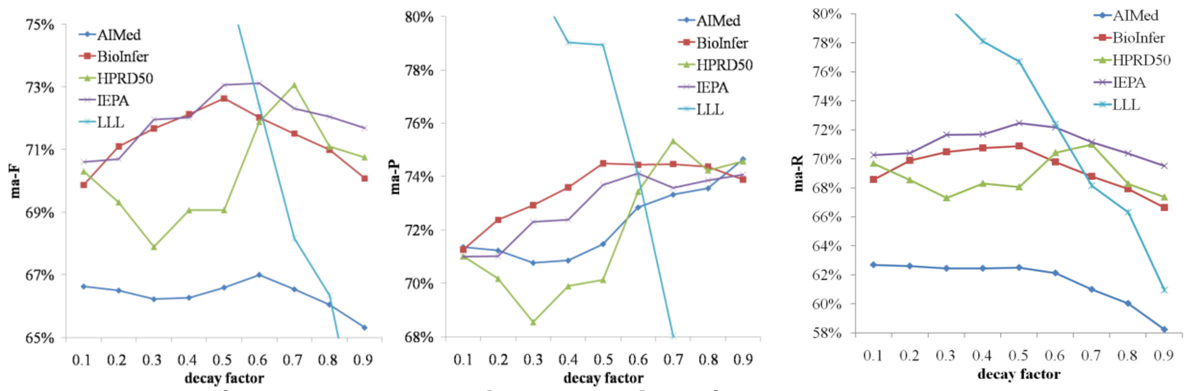


Figure 5. Performance variation with respect to decay factor in Five PPI Corpora. Macro-averaged F1 (left), Precision (middle), Recall (right) evaluated by 10-fold CV

Training corpora	Systems	*F*-Scores in the test corpora				
		AIMed	BioInfer	HPRD50	IEPA	LLL
AIMed	Our System	**67.0**	**64.2**	**72.9**	59.0	62.7
	(Miwa et al., 2009a)	60.8	53.1	68.3	**68.1**	73.5
	(Airola et al., 2008)	56.4	47.1	69.0	67.4	**74.5**
BioInfer	Our System	**65.2**	**72.6**	**71.9**	**72.9**	**78.4**
	(Miwa et al., 2009a)	49.6	68.1	68.3	71.4	76.9
	(Airola et al., 2008)	47.2	61.3	63.9	68.0	78.0
HPRD50	Our System	**63.1**	**65.5**	**73.1**	**69.3**	**73.7**
	(Miwa et al., 2009a)	43.9	48.6	70.9	67.8	72.2
	(Airola et al., 2008)	42.2	42.5	63.4	65.1	67.9
IEPA	Our System	**57.8**	**66.1**	66.3	73.1	78.4
	(Miwa et al., 2009a)	40.4	55.8	66.5	71.7	**83.2**
	(Airola et al., 2008)	39.1	51.7	**67.5**	**75.1**	77.6
LLL	Our System	**55.9**	**64.4**	**69.4**	**71.4**	82.1
	(Miwa et al., 2009a)	38.6	48.9	64.0	65.6	**83.2**
	(Airola et al., 2008)	33.3	42.5	59.8	64.9	76.8

Table 3. Macro-averaged *F*1 scores in cross-corpora evaluation. Rows and columns correspond to the training and test corpora, respectively. We parallel our results with other recently reported results. All the split methods in 10-fold CV are the same for fair comparisons.

As seen in the table, the proposed method is superior to all the others in *F*-scores. The improvement in precision (12.8%) is most significant, especially in comparison with the work of Miwa et al., (2009b), which used multiple corpora (AIMed + IEPA) for training and combined various kernels such as bag-of-words, parse trees and graphs. It is natural that the recall value is lower since a less number of patterns (features) must have been learned. What's important is that the proposed method has a higher or at least comparable overall performance without additional resources.

Our approach is significantly better than that of Airola et al., (2008), which employed two different forms of graph kernels to improve the initial model. Since they did not use multiple corpora for training, the comparison shows the direct benefit of using the extension of the kernel.

6 Conclusion and Future Works

To improve the performance of PPIE, recent research activities have had a tendency of increasing the complexity of the systems by combining various methods and resources. In this paper, however, we argue that by paying more

	POS	NEG	*ma*-P	*ma*-R	*ma*-F	σ_F
Our System	1,000	4,834	**72.8**	62.1	**67.0**	**4.5**
(Miwa et al., 2009b)	1,000	4,834	60.0	**71.9**	65.2	
(Miwa et al., 2009a)	1,000	4,834	58.7	66.1	61.9	7.4
(Miwa et al., 2008)	1,005	4,643	60.4	69.3	61.5	
(Miyao et al., 2008)	1,059	4,589	54.9	65.5	59.5	
(Giuliano et al., 2006)	-	-	60.9	57.2	59.0	
(Airola et al., 2008)	1,000	4,834	52.9	61.8	56.4	5.0
(Sætre et al., 2007)	1,068	4,563	64.3	44.1	52.0	
(Erkan et al., 2007)	951	4,020	59.6	60.7	60.0	
(Bunescu & Mooney, 2005)	-	-	65.0	46.4	54.2	

Table 4. Comparative results in AIMed. The number of positive instances (POS) and negative instances (NEG) and macro-averaged precision (*ma*-P), recall (*ma*-R) and *F*1-score (*ma*-F) are shown.

attention to a single model and adjusting parameters more carefully, we can obtain at least comparable performance if not better.

This paper indicates that a well-tuned parse tree kernel based on decay factor can achieve the superior performance in PPIE when it is preprocessed by the path-enclosed tree pruning method. It was shown in a series of experiments that our system produced the best scores in single corpus evaluation as well as cross-corpora validation in comparison with other state-of-the-art methods. Contribution points of this paper are as follows:

(1) We have shown that the benefits of using additional resources including richer features can be obtained by tuning a single tree kernel method with tree pruning and decaying factors.

(2) We have newly found that the decay factor influences precision enhancement of PPIE and hence its overall performance as well.

(3) We have also revealed that the parse tree kernel method equipped with decay factors shows superior generalization power even with small corpora while presenting significant performance increase on cross-corpora experiments.

As a future study, we leave experiments with training the classifier with multiple corpora and deeper analysis of what aspects of the corpora gave different magnitudes of the improvements.

Acknowledgment

We want to thank the anonymous reviewers for their valuable comments. This work has been supported in part by KRCF Grant, the Korean government.

Reference

Airola, A., Pyysalo, S., Bjorne, J., Pahikkala, T., Ginter, F. & Salakoski, T. (2008). All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. BMC Bioinformatics, 9(S2), doi:10.1186/1471-2105-9-S11-S2.

Andrade, M. A. & Valencia, A. (1998). Automatic extraction of keywords from scientific text: application to the knowledge domain of protein families. Bioinformatics, 14(7), 600-607.

Blaschke, C., Andrade, M., Ouzounis, C. & Valencia, A. (1999). Automatic extraction of biological information from scientific text: protein-protein interactions. Proc. Int. Conf. Intell. Syst. Mol. Biol., (pp. 60-67).

Bunescu, R., Ge, R., Kate, R., Marcotte, E., Mooney, R., Ramani, A. & Wong, Y. (2005). Comparative Experiments on Learning Information Extractors for Proteins and their Interactions. Artif. Intell. Med., Summarization and Information Extraction from Medical Documents, 33, 139-155

Collins, M. & Duffy, N. (2001). Convolution Kernels for Natural Language. NIPS-2001, (pp. 625-632).

Craven, M. & Kumlien, J. (1999). Constructing biological knowledge bases by extracting information from text sources. Proceedings of the 7th International conference on intelligent systems for molecular biology, (pp.77-86), Heidelberg, Germany.

Ding, J., Berleant, D., Nettleton, D. & Wurtele, E. (2002). Mining MEDLINE: abstracts, sentences, or phrases?. Proceedings of PSB'02, (pp. 326-337)

Erkan, G., Ozgur, A., & Radev, D. R., (2007). Semi-supervised classification for extracting protein interaction sentences using dependency parsing. In EMNLP 2007.

Fundel, K., Küffner, R. & Zimmer, R. (2007). RelEx – Relation extraction using dependency parse trees. Bioinformatics, 23, 365-371.

Giuliano, C., Lavelli, A., Romano, L., (2006). Exploiting Shallow Linguistic Information for Relation Extraction From Biomedical Literature. Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics.

Gondy, L., Hsinchun C. & Martinez Jesse D. (2003). A shallow parser based on closed-class words to capture relations in biomedical text. J. Biomed. Informatics. 36(3), 145-158.

GuoDong, Z., Min, Z., Dong, H. J. & QiaoMing, Z. (2007). Tree Kernel-based Relation Extraction with Context-Sensitive Structured Parse Tree Information. Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, (pp. 728–736)

Marcotte, E. M., Xenarios, I. & Eisenberg D. (2001). Mining literature for protein-protein interactions. Bioinformatics, 17(4), 359-363.

Miwa, M., Sætre, R., Miyao, Y. & Tsujii J. (2009a). Protein-protein interaction extraction by leveraging multiple kernels and parsers. International Journal of Medical Informatics, 78(12), e39-e46.

Miwa, M., Sætre, R., Miyao, Y. & Tsujii J. (2009b). A Rich Feature Vector for Protein-Protein Interaction Extraction from Multiple Corpora. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, (pp. 121-130)

Miwa, M., Sætre, R., Miyao, Y., Ohta, T., & Tsujii, J. (2008). Combining multiple layers of syntactic information for protein-protein interaction extraction. In Proceedings of the Third International Symposium on Semantic Mining in Biomedicine (SMBM 2008), (pp. 101–108)

Miyao, Y., Sætre, R., Sagae, K., Matsuzaki, T., & Tsujii, J. (2008). Task-oriented evaluation of syntactic parsers and their representations. Proceedings of the 45th Meeting of the Association for Computational Linguistics (ACL'08:HLT).

Moschitti, A. (2006). Making tree kernels practical for natural language learning. Proceedings of EACL'06, Trento, Italy.

Nikolai, D., Anton, Y., Sergei, E., Svetalana, N., Alexander, N. & llya, M. (2004). Extracting human protein interactions from MEDLINE using a full-sentence parser. Bioinformatics, 20(5), 604-611.

Ono, T., Hishigaki, H., Tanigam, A. & Takagi, T. (2001). Automated extraction of information on protein-protein interactions from the biological literature. Bioinformatics, 17(2), 155-161.

Pyysalo, S., Airola, A., Heimonen, J., Björne, J., Ginter, F. & Salakoski, T. (2008). Comparative analysis of five protein-protein interaction corpora. BMC Bioinformatics, 9(S6), doi:10.1186/1471-2105-9-S3-S6.

Sætre, R., Sagae, K., & Tsujii, J. (2007). Syntactic features for protein-protein interaction extraction. In LBM 2007 short papers.

Sekimizu, T., Park H. S. & Tsujii J. (1998). Identifying the interaction between genes and gene products based on frequently seen verbs in MEDLINE abstracts. Workshop on genome informatics, vol. 9, (pp. 62-71).

Shawe-Taylor, J., Cristianini, N., (2004). Kernel Methods for Pattern Analysis, Cambridge University Press.

Temkin, J. M. & Gilder, M. R. (2003). Extraction of protein interaction information from unstructured text using a context-free grammar. Bioinformatics, 19(16), 2046-2053.

Vishwanathan, S. V. N., Smola, A. J. (2003). Fast Kernels for String and Tree Matching. Advances in Neural Information Processing Systems, 15, 569-576, MIT Press.

Zhang, M., GuoDong, Z. & Aiti, A. (2008). Exploring syntactic structured features over parse trees for relation extraction using kernel methods. Information Processing and Management, 44, 687-701

Zhang, M., Zhang, J., Su, J. & Zhou, G. (2006). A Composite Kernel to Extract Relations between Entities with both Flat and Structured Features. 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL, (pp.825-832).

Zhou, D. & He, Y. (2008). Extracting interactions between proteins from the literature. Journal of Biomedical Informatics, 41, 393-407.

An ontology-driven system for detecting global health events

Nigel Collier
National Inst. Informatics
collier@nii.ac.jp

Reiko Matsuda Goodwin
Fordham University
reikogoodwin@gmail.com

John McCrae
Bielefeld University
johnmccrae@gmail.com

Son Doan
Vanderbilt University
son.doan@vanderbilt.edu

Ai Kawazoe
Tsuda College
zoeai@tsuda.ac.jp

Mike Conway
University of Pittsburgh
conwaym@pitt.edu

Asanee Kawtrakul
Kasetart University
ak@ku.ac.th

Koichi Takeuchi
Okayama University
koichi@cs.okayama-u.ac.jp

Dinh Dien
VietNam National University
ddien66@yahoo.com

Abstract

Text mining for global health surveillance is an emerging technology that is gaining increased attention from public health organisations and governments. The lack of multilingual resources such as WordNets specifically targeted at this task have so far been a major bottleneck. This paper reports on a major upgrade to the BioCaster Web monitoring system and its freely available multilingual ontology; improving its original design and extending its coverage of diseases from 70 to 336 in 12 languages.

1 Introduction

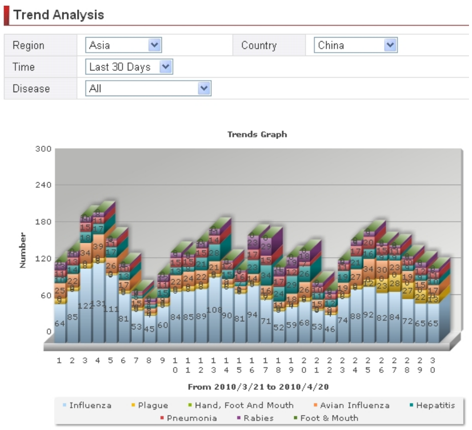
The number of countries who can sustain teams of experts for global monitoring of human/animal health is limited by scarce national budgets. Whilst some countries have advanced sensor networks, the world remains at risk from the health impacts of infectious diseases and environmental accidents. As seen by the recent A(H5N1), A(H1N1) and SARS outbreaks, a problem in one part of the world can be rapidly exported, leading to global hardship.

The World Health Organization (WHO) estimates that in the future, between 2 to 7.4 million people could be at risk worldwide from a highly contageous avian flu virus that spreads rapidly through the international air travel network (WHO, 2005). Pandemics of novel pathogens have the capacity to overwhelm healthcare systems, leading to widespread morbidity, mortality and socio-economic disruption (Cox et al., 2003). Furthermore, outbreaks of livestock diseases, such as foot-and-mouth disease or equine influenza can have a devastating impact on industry, commerce and human health (Blake et al., 2003). The challenge is to enhance vigilance and control the emergence of outbreaks. Whilst human analysis remains essential to spot complex relationships, automated analysis has a key role to play in filtering the vast volume of data in real time and highlighting unusual trends using reliable predictor indicators.

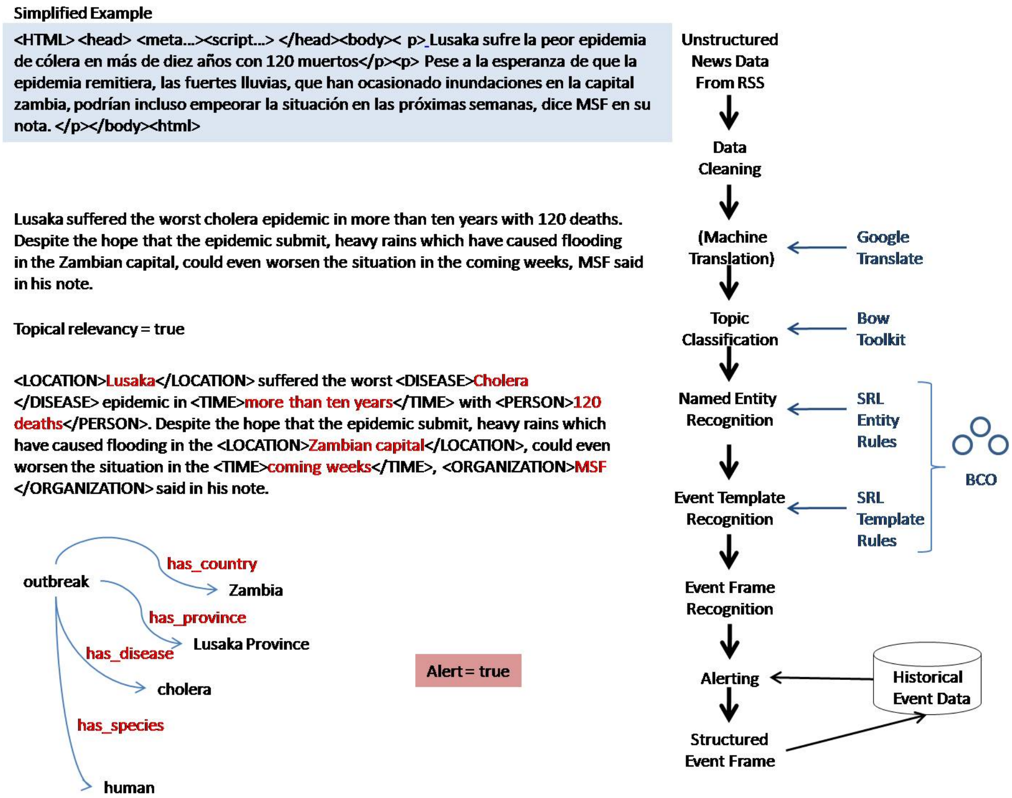
BioCaster (http://born.nii.ac.jp) (Collier et al., 2008) is a Web 2.0 monitoring station for the early detection of infectious disease events. The system exploits a high-throughput semantic processing pipeline, converting unstructured news texts to structured records, alerting events based on time-series analysis and then sharing this information with users via geolocating maps (Fig. 1(a)), graphs (Fig. 1(b)) and alerts. Underlying the system is a publicly available multilingual application ontology. Launched in 2006 (Collier et al., 2006) the BioCaster Ontology (BCO) has been downloaded by over 70 academic and industrial groups worldwide. This paper reports on a major upgrade to the system and the ontology - expanding the number of languages from 6 to 12, redefining key relations and extending coverage in the number of diseases from 70 to 336, including many veterinary diseases.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 215–222,
Beijing, August 2010

(a) Bio-geographic map



(b) Trend graph analyser



(c) BioCaster processes

Figure 1: (a)BioCaster's bio-geographic map for a suspected foot-and-mouth outbreak on 22^{nd} March, 2010 with links to the multilingual ontology, NCBI, HighWire, GoPubMed and Google Scholar; (b) The trends analyser showing aggregated document counts for health events in China between 13^{nd} March and 12^{th} April, 2010; (c) The system's pipeline of processes with example semantic markup.

2 Background

As the world becomes more interconnected and urbanized and animal production becomes increasingly intensive, the speed with which epidemics spread becomes faster, adding to pressure on biomedical experts and governments to make quick decisions. Traditional validation methods such as field investigations or laboratory analysis are the mainstay of public health but can require days or weeks to issue reports. The World Wide Web with its economical and real time delivery of information represents a new modality in health surveillance (Wagner and Johnson, 2006) and has been shown to be an effective source by the World Health Organization (WHO) when Public Health Canada's GPHIN system detected the SARS outbreak in southern China from news reports during November 2002. The recent A(H1N1) 'swine flu' pandemic highlighted the trend towards agencies using unvalidated sources. The technological basis for such systems can be found in statistical classification approaches and light weight ontological reasoning. For example, Google Flu Trends (Ginsberg et al., 2009) is a system that depends almost entirely on automatic statistical classification of user queries; MedISys-PULS (Yangarber et al., 2008), HealthMap (Freifeld et al., 2008) and BioCaster use a mixture of statistical and ontological classification; and GPHIN (Mawudeku and Blench, 2006) and Argus (Wilson, 2007) rely on a mixture of ontological classification and manual analysis.

Compared to other similar systems BioCaster is characterized by its richly featured and publicly downloadable ontology and emphasizes critical evaluation of its text mining modules. Empirical results have included: topic classification, named entity recognition, formal concept analysis and event recognition. In the absence of a community gold standard, task performance was assessed on the best available 'silver' standard - the ProMED-mail network (Madoff and Woodall, 2005), achieving F-score of 0.63 on 14 disease-country pairs over a 365-day period (Collier, 2010).

Despite initial skepticism within the public health community, health surveillance systems based on NLP-supported human analysis of media reports are becoming firmly established in Europe, North America and Japan as sources of health information available to governments and the public (Hartley et al., 2010). Whilst there is no substitute for trained human analysts, automated filtering has helped experts save time by allowing them to sift quickly through massive volumes of media data. It has also enabled them to supplement traditional sources with a broader base of information.

In comparison with other areas of biomedical NLP such as the clinical and genetics' domains, a relative lack of building block resources may have hindered the wider participation of NLP groups in public health applications. It is hoped that the provision of common resources like the BCO can help encourage further development and benchmarking.

3 Method

BioCaster performs analysis of over 9000 news articles per day using the NPACI Rocks cluster middleware (http://www.rockcsclusters.org) on a platform of 48 3.0GHz Xeon cores. Data is ingested 24/7 into a semantic processing pipeline in a short 1 hour cycle from over 1700 public domain RSS feeds such as Google news, the European Media Monitor and ProMED-mail. Since 2009, news has also being gathered under contract from a commercial news aggregation company, providing access to over 80,000 sources across the world's languages.

The new 2010 version of BioCaster uses machine translation into English (eleven languages) to source news stories related to currently occurring infectious and environmental disease outbreaks in humans, animals and plants.

Access to the site is freely available but login registration applies to some functions such as email alerts. Processing is totally automatic, but we have the potential within the login system to enable human moderated alerts which broadcast to Twitter and RSS.

Below we describe in detail two key aspects of the system that have been significantly upgraded: the BCO and the event detection system.

3.1 Ontology

3.1.1 Aim

The BioCaster Ontology aims:

- To describe the terms and relations necessary to detect and risk assess public health events in the grey literature;
- To bridge the gap between (multilingual) grey literature and existing standards in biomedicine;
- To mediate integration of content across languages;
- To be freely available.

The central knowledge source for BioCaster is the multilingual ontology containing domain terms such as diseases, agents, symptoms, syndromes and species as well as domain sensitive relations such as a disease causing symptoms or an agent affecting particular host species. This allows the text mining system to have a basic understanding of the key concepts and relationships within the domain to fill in gaps not mentioned explicitly in the news reports. To the best of our knowledge the BCO is unique as an application ontology, providing freely available multilingual support to system developers interested in outbreak surveillance in the language of the open media.

The BCO however has little to say outside of its application domain, e.g. in disease-gene interaction or for supporting automatic diagnosis. As discussed in Grey Cowell and Smith (2010), there are many other resources available that have the potential to support applications for infectious disease analysis including controlled vocabularies and ontologies such as the the Unified Medical Language System (UMLS) (Lindberg et al., 1993), International Classification of Diseases (ICD-10) (WHO, 2004), SNOMED CT (Stearns et al., 2001), Medical Subject Headings (MeSH) (Lipscomb, 2000) and the Infectious Disease Ontology (IDO) (Grey Cowell and Smith, 2010). In (Collier et al., 2006) we discussed how BCO compared to such ontologies so we will focus from now on the implication of the extensions.

3.1.2 Scope

The new version of the BCO now covers 12 languages including all the United Nation's official languages: Arabic (968 terms), English (4113), French (1281), Indonesian (1081), Japanese (2077), Korean (1176), Malaysian (1001), Russian (1187), Spanish (1171), Thai (1485), Vietnamese (1297) and Chinese (1142). The multilingual ontology can be used as a direct knowledge source in language-specific text mining modules, as an indexing resource for searching across concepts in various languages and as a dictionary for future translation modules. Currently news in all 12 languages is available via the Web portal but news in additional languages such as German, Italian and Dutch are being added using machine translation.

3.1.3 Design

Like EuroWordNet (Vossen, 1998), on which it is loosely based, the BCO adopts a thesaurus-like structure with synonym sets linking together terms across languages with similar meaning. Synonym sets are referred to using *root terms*. Root terms themselves are fully defined instances that provide bridges to external classification schemes and nomenclatures such as ICD10, MeSH, SNOMED CT and Wikipedia. The central backbone taxonomy is deliberately shallow and taken from the ISO's Suggested Upper Merged Ontology (Niles and Pease, 2001). To maintain consistency and computability we kept a single inheritance structure throughout. 18 core domain concepts corresponding to named entities in the text mining system such as DISEASE and SYMPTOM were the results of analysis using a formal theory (Guarino and Welty, 2000).

We have endeavoured to construct definitions for root terms along Aristotelean principles by specifying the difference to the parent. For example in the case of *Eastern encephalitis virus*:

> *Eastern equine encephalitis virus is a species of virus that belongs to the genus Alphavirus of the family Togaviridae (order unassigned) of the group IV ((+)ssRNA) that possesses a positive single stranded RNA genome. It is the*

etiological agent of the eastern equine encephalitis.

We are conscious though that terms used in the definitions still require more rigorous control to be considered useful for machine reasoning. To aid both human and machine analysis root terms are linked by a rich relational structure reflecting domain sensitive relations such as *causes(virus,disease)*, *has_symptom(disease, symptom)*, *has_associated_syndrome(disease, syndrome)*, *has_reservoir(virus, organism)*.

In such a large undertaking, the order of work was critical. We proceeded by collecting a list of notifiable diseases from national health agencies and then grouped the diseases according to perceived relevance to the International Health Regulations 2005 (Lawrence and Gostin, 2004). In this way we covered approximately 200 diseases, and then explored freely available resources and the biomedical literature to find academic and layman's terminology to describe their agents, affected hosts, vector species, symptoms, etc. We then expanded the coverage to less well known human diseases, zoonotic diseases, animal diseases and diseases caused by toxic substances such as sarin, hydrogen sulfide, sulfur dioxide and ethylene. At regular stages we checked and validated terms against those appearing in the news media.

As we expanded the number of conditions to include veterinary diseases we found a major structural reorganization was needed to support animal symptoms. For example, a high temperature in humans would not be the same as one in bovids. This prompted us in the new version to group diseases and symptoms around major animal familes and related groups, e.g. *high temperature (human)* and *high temperature (bovine)*.

A second issue that we encountered was the need to restructure the hierarchy under *OrganicObject* which was divided between *MicroOrganism* and *Animal*. The structure of the previous version meant that the former were doing double duty as infecting agents and the later were affected hosts. The *MicroOrganism* class contained bacterium, helminth, protozoan, fungus and virus, which then became the domain in a relation 'x causes y'. Expansion forced us to accomodate the fact that some animals such as worms and mites (e.g. scabies) also infect humans as well as animals. The result was a restructuring of the organic classes using the Linnean taxonomy as a guideline, although this is probably not free from errors (e.g. virus is typically not considered to be an organism).

3.2 Event alerting system

Figure 1(c) shows a schematic of the modular design used by the BioCaster text mining system. Following on from machine translation and topic classification is named entity recognition and template recognition which we describe in more detail below. The final structured event frames include slot values normalized to ontology root terms for disease, pathogen (virus or bacterium), country and province. Additionally we also identify 15 aspects of public health events critical to risk assessment such as: spread across international borders, hospital worker infection, accidental or deliberate release, food contamination and vaccine contamination.

Latitude and longitude of events down to the province level are found in two ways: using the Google API up to a limit of 15000 lookups per day, and then using lookup on the BCO taxonomy of 5000 country and province names derived from open sources such as Wikipedia.

Each hour events are automatically alerted to a Web portal page by comparing daily aggregated event counts against historical norms (Collier, 2010). Login users can also sign up to receive emails on specific topics. A topic would normally specify a disease or syndrome, a country or region and a specific risk condition.

In order to extract knowledge from documents, BioCaster maintains a collection of rule patterns in a regular expression language that converts surface expressions into structured information. For example the surface phrase "man exposes airline passengers to measles" would be converted into the three templates "**species(human); disease(measles); international_travel(true)**". Writing patterns to produce such templates can be very time consuming and so the BioCaster project has developed its own

D3: :- name(disease){ list(@undiagnosed) words(,1) list(@disease) }
S2: :- name(symptom) { list(@severity) list(@symptom)}
CF1: contaminated_food("true") :- "caused" "by" list(@contaminate_verbs_past) list(@injested_material)
SP4: species("animal") :- name(animal,A) words(,3) list(@cull_verbs_past)

Table 1: Examples of SRL rules for named entity and template recognition. Template rules contain a label, a head and a body, where the head specifies the template pattern to be output if the body expression matches. The body can contain word lists, literals, and wild cards. Various conditions can be placed on each of these such as orthographic matching.

light weight rule language - called the Simple Rule Language (SRL) and a pattern building interface for maintaining the rule base (McCrae et al., 2009). Both are freely available to the research community under an open source license. Currently BioCaster uses approximately 130 rules for entity recognition, 1000 word lists and 3200 template rules (of which half are for location recognition) to identify events of interest in English. Using SRL allows us to quickly adapt the system to newly emerging terminology such as the 11+ designations given to A(H1N1) during the first stages of the 2009 pandemic.

The SRL rulebook for BioCaster can recognize a range of entities related to the task of disease surveillance such as bacteria, chemicals, diseases, countries, provinces, cities and major airports. Many of these classes are recognized using terms imported from the BCO. The rule book also contains specialised thesauri to recognize subclasses of entities such as locations of habitation, eateries and medical service centres. Verb lists are maintained for lexical classes such as detection, mutation, investigation, causation, contamination, culling, blaming, and spreading.

Some examples of SRL rules for named entity recognition are shown in Table 1 and described below:

Rule D3 in the rulebook tags phrases like 'mystery illness' or 'unknown killer bug' by matching on strings contained within two wordlists, @undiagnosed and @disease, separated by up to one word.

Rule S2 allows severity indicators such as 'severe' or 'acute' to modify a list of known symptoms in order to identify symptom entities.

Rule CF1 is an example of a template rule. If the body of the rule matches by picking out expressions such as 'was caused by tainted juice', this triggers the head to output an alert for contaminated food.

Rule SP4 identifies the victim species as 'animal' in contexts like '250 geese were destroyed'.

The rulebook also supports more complex inferences such as the home country of national public health organizations.

Since BioCaster does not employ systematic manual checking of its reports, it uses a number of heuristic filters to increase specificity (the proportion of correctly identified negatives) for reports that appear on the public Web portal pages. For example, reports with no identified disease and country are rejected. Since these heuristics may reduce sensitivity they are not applied to news that appears on the user login portal pages.

4 Results and Discussion

Version 3 of the ontology represents a significant expansion in the coverage of diseases, symptoms and pathogens on version 2. Table 2 summarizes the number of root terms for diseases classified by animal familes.

The thesaurus like structure of the BCO is compatible in many respects to the Simple Knowledge Organization System (SKOS) (Miles et al., 2005). In order to extend exchange and re-use we have produced a SKOS version of the BCO which is available from the BCO site. We have also converted the BCO terms into 12 SRL rule books (1 for each language) for entity tagging. These too are freely available from the BCO site.

As the ontology expands we will consider adopting a more detailed typing of diseases such as *hasInfectingPart* to indicate the organ affected

Species	N	Example
Avian	22	Fowl pox
Bee	6	Chalk brood disease
Bovine	24	Bluetongue
Canine	4	Blastomycosis (Canine)
Caprine	14	Contagious agalactia
Cervine	2	Chronic wasting disease
Equine	17	Strangles
Feline	4	Feline AIDS
Fish	2	Viral hemorr hagic septicemia
Human	216	Scarlet fever
Lagomorph	2	Myxomatosis
Non-human primate	16	Sylvan yellow fever
Other	2	Crayfish plague
Rodent	8	Colorado tick fever (Rodent)
Swine	12	Swine erysipelas

Table 2: Major disease groups organized by affected animal family. N represents the number of root terms.

or *hasProtectionMethod* to indicate broad classes of methods used to prevent or treat a condition. The typology of diseases could also be extended in a more fine grained manner to logically group conditions, e.g. *West Nile virus encephalitis*, *Powassan encephalitis* and the *Japanese B encephalitis* could be connected through a *hasType* relation on *encephalitis*.

5 Conclusion

Multilingual resources specifically targeted at the task of global health surveillance have so far been very rare. We hope that the release of version 3 can be used to support a range of applications such as text classification, cross language search, machine translation, query expansion and so on.

The BCO has been constructed to provide core vocabulary and knowledge support to the BioCaster project but it has also been influential in the construction of other public health oriented application ontologies such as the Syndromic Surveillance Ontology (Okhamatovskaia et al., 2009). The BCO is freely available from http://code.google.com/p/biocaster-ontology/ under a Creative Commons license.

Acknowledgements

The authors greatly acknowledge the many coworkers who have provided comments and feedback on BioCaster. Funding support was provided in part by the Japan Science and Technology Agency under the PRESTO programme.

References

Blake, A., M. T. Sinclair, and G. Sugiyarto. 2003. Quantifying the impact of foot and mouth disease on tourism and the UK economy. *Tourism Economics*, 9(4):449–465.

Collier, N., A. Kawazoe, L. Jin, M. Shigematsu, D. Dien, R. Barrero, K. Takeuchi, and A. Kawtrakul. 2006. A multilingual ontology for infectious disease surveillance: rationale, design and challenges. *Language Resources and Evaluation*, 40(3–4). DOI: 10.1007/s10579-007-9019-7.

Collier, N., S. Doan, A. Kawazoe, R. Matsuda Goodwin, M. Conway, Y. Tateno, Q. Ngo, D. Dien, A. Kawtrakul, K. Takeuchi, M. Shigematsu, and K. Taniguchi. 2008. BioCaster:detecting public health rumors with a web-based text mining system. *Bioinformatics*, 24(24):2940–1, December. doi:10.1093/bioinformatics/btn534.

Collier, N. 2010. What‘s unusual in online disease outbreak news? *Biomedical Semantics*, 1(1), March. doi:10.1186/2041-1480-1-2.

Cox, N., S. Temblyn, and T. Tam. 2003. Influenza pandemic planning. *Vaccine*, 21(16):1801–1803.

Freifeld, C., K. Mandl, B. Reis, and J. Brownstein. 2008. Healthmap: global infectious disease monitoring through automated classification and visualization of internet media reports. *J. American Medical Informatics Association*, 15:150–157.

Ginsberg, J., M. Mohebbi, R. Patel, L. Brammer, M. Smolinski, and L. Brilliant. 2009. Detecting influenza epidemics using search engine query data. *Nature*, 457:1012–1014.

Grey Cowell, L. and B. Smith. 2010. Infectious disease informatics. In Sintchenko, V., editor, *Infectious Disease Informatics*, pages 373–395. Springer New York.

Guarino, N. and C. Welty. 2000. A formal ontology of properties. In Dieng, R. and O. Corby, editors, *EKAW-2000: Proc. 12th Int. Conf. on Knowledge Engineering and Knowledge Management*, pages 97–112.

Hartley, D., N. Nelson, R. Walters, R. Arthur, R. Yangarber, L. Madoff, J. Linge, A. Mawudeku, N. Collier, J. Brownstein, G. Thinus, and N. Lightfoot. 2010. The landscape of international biosurveillance. *Emerging Health Threats J.*, 3(e3), January. doi:10.1093/bioinformatics/btn534.

Lawrence, O. and J. Gostin. 2004. International infectious disease law - revision of the World Health Organization's international health regulations. *J. American Medical Informatics Association*, 291(21):2623–2627.

Lindberg, Donald A.B., L. Humphreys, Betsy, and T. McCray, Alexa. 1993. The unified medical language system. *Methods of Information in Medicine*, 32:281–291.

Lipscomb, C. 2000. Medical subject headings (MeSH). *Bulletin of the Medical Library Assocation*, 88:256–266.

Madoff, Lawrence C. and John P. Woodall. 2005. The internet and the global monitoring of emerging diseases: Lessons from the first 10 years of promedmail. *Archives of Medical Research*, 36(6):724 – 730. Infectious Diseases: Revisiting Past Problems and Addressing Future Challenges.

Mawudeku, A. and M. Blench. 2006. Global public health intelligence network (gphin). In *Proc. 7th Int. Conf. of the Association for Machine Translation in the Americas, Cambridge, MA, USA*, August 8–12.

McCrae, J., M. Conway, and N. Collier. 2009. Simple rule language editor. Google code project, September. Available from: http://code.google.com/p/srl-editor/.

Miles, A., B. Matthews, and M. Wilson. 2005. SKOS Core: Simple knowledge organization for the web. In *Proc. Int. Conf. on Dublin Core and Metadata Applications, Madrid, Spain*, 12–15 September.

Niles, I. and A. Pease. 2001. Towards a standard upper ontology. In Welty, C. and B. Smith, editors, *2nd Int. Conf. on Formal Ontology in Information Systems FOIS-2001, Maine, USA*, October 17–19.

Okhamatovskaia, A., W. Chapman, N. Collier, J. Espino, and D. Buckeridge. 2009. SSO: The syndromic surveillance ontology. In *Proc. Int. Soc. for Disease Surveillance, Miami, USA*, December 3–4.

Stearns, M. Q., C. Price, K. A. Spackman, and A. Y. Wang. 2001. SNOMED clinical terms: overview of the development process and project status. In *Proc. American Medical Informatics Association (AMIA) Symposium*, pages 662–666.

Vossen, P. 1998. Introduction to EuroWordNet. *Computers and the Humanities*, 32:73–89.

Wagner, M. and H. Johnson. 2006. The internet as sentinel. In Wagner, M. et al., editor, *The Handbook of Biosurveillance*, pages 375–385. Academic Press.

WHO. 2004. *ICD-10, International Statistical Classification of Diseases and Related Health Problems, Tenth Revision*. World Health Organization, December.

WHO. 2005. Avian influenza: assessing the pandemic threat. Technical Report WHO/CDS/2005.29, World Health Organization, Geneva, January.

Wilson, J. 2007. Argus: a global detection and tracking system for biological events. *Advances in Disease Surveillance*, 4.

Yangarber, R., P. von Etter, and R. Steinberger. 2008. Content collection and analysis in the domain of epidemiology. In *Proc. Int. Workshop on Describing Medical Web Resources (DRMED 2008), Gotenburg, Sweden*, May 27th.

Constraining robust constructions for broad-coverage parsing with precision grammars

Bart Cramer[†] **and Yi Zhang**[†‡]
Department of Computational Linguistics & Phonetics, Saarland University[†]
LT-Lab, German Research Center for Artificial Intelligence (DFKI)[‡]
{bcramer,yzhang}@coli.uni-saarland.de

Abstract

This paper addresses two problems that commonly arise in parsing with precision-oriented, rule-based models of grammar: lack of speed and lack of robustness. First, we show how we can reduce parsing times by restricting the number of tasks the parser will carry out, based on a generative model of rule applications. Second, we show that a combination of search space restriction and radically overgenerating robustness rules lead to a more robust parser, with only a small penalty in precision. Applying both the robustness rules and a fragment fallback strategy showed better recall than just giving fragment analyses, with equal precision. Results are reported on a medium-sized HPSG grammar for German. [1]

1 Introduction

In the field of natural language processing, it is common wisdom that handwritten, rule-based models generally perform poorly on complex problems, mainly due to the knowledge acquisition bottleneck: it is hard for the human modeller to conceive of all possible scenarios the model has to cope with. In parsing, many approaches have relied on hand-written grammars, and their fragility is one of their largest weaknesses. Such models can fail due to insufficiency of lexical entries or grammatical constructions, but also due to creative or ungrammatical input. In any case, the parser should always return a reasonable output. A very simple technique is partial or fragment parsing (Kiefer et al., 1999; Riezler et al., 2001; Zhang et al., 2007a): if there is no item in the chart that both spans the complete sentence and fulfills the root condition, several chunks that do conform to a root condition are combined by minimising a certain cost function (for instance to favour larger chunks, or more probable chunks).

A second problem with deep parsers is their relatively low efficiency. For online applications, it is impermissible to wait for longer than a minute before the system responds. Apart from studies that were aimed at increasing the efficiency of deep parsers by using smarter algorithms (e.g. using left-corner relations (Van Noord, 1997)), several studies in recent years have suggested that search space restriction can offer a beneficial balance between speed and accuracy as well. Techniques that have been proposed are, among others, supertagging (Clark and Curran, 2007), CFG filtering (Matsuzaki et al., 2007) and beam thresholding (Ninomiya et al., 2005).

A potential disadvantage of the latter technique is that the unifications have taken place by the time the value of the resulting chart item is investigated. One strategy that tries to prevent execution of unlikely tasks altogether is presented by van Noord (2009). In this method, the parser learns from an unannotated corpus which parse steps contributed to the solution as preferred by the disambiguation model (as opposed to a certain gold standard). Hence, this approach is self-learning.

Another study that is close to our approach

[1] The research reported on in this paper has been carried out with financial support from the Deutsche Forschungsgemeinschaft and the German Excellence Cluster of Multimodal Computing & Interaction.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 223–231,
Beijing, August 2010

to search space restriction is c-structure pruning (Cahill et al., 2008). The authors show that a large, hand-written, unification-based parser (the XLE LFG parser for English) can perform reasonably faster (18%) without losing accuracy, by not allowing the parser to unify if the resulting item will have a span that does not conform to a CFG tree that was generated from the sentence beforehand by a PCFG parser. Much better results (67% speed-up) are obtained by pruning chart items locally, based on their relative probabilities (Cahill et al., 2008). This is the approach that is closest to the one we present in this paper.

In this paper, we introduce a method that addresses robustness and efficiency concurrently. The search space is restricted by setting a maximum on the number of tasks per chart cell. Because tasks are carried out according to a priority model based on the generative probabilities of the rule applications, it is unlikely that good readings are dropped. More robustness is achieved by adding radically overgenerating rules to the grammar, which could cover all sentences, given an disproportionate amount of time and memory. By strongly restricting the search space, however, the computation requirements remains within bounds. Because the robustness rules are strongly dispreferred by both the priority model and the disambiguation model, all sentences that would be covered by the 'restricted' grammar remain high-precision, but sentences that are not covered will get an additional push from the robustness rules.

1.1 An HPSG grammar for German

The grammar we use (Cramer and Zhang, 2009) is the combination of a hand-written, constraint-based grammar in the framework of HPSG and an open word class lexicon extracted from the Tiger treebank (Brants et al., 2002) in a deep lexical acquisition step. One of the aims of this grammar is to be precision-oriented: it tries to give detailed analyses of the German language, and reject ungrammatical sentences as much as possible. However, this precision comes at the cost of lower coverage, as we will see later in this paper.

Along with the grammar, a treebank has been developed by re-parsing the Tiger treebank, and including those sentences for which the grammar was able to reproduce the original Tiger dependencies. The treebank's size is just over 25k sentences (only selected from the first 45k sentences, so they don't overlap with either the development or test set), and contains the correct HPSG derivation trees. These (projective) derivation trees will function as the training set for the statistical models we develop in this study.

2 Restriction of the search space

2.1 The PET parser

The parser we employ, the PET parser (Callmeier, 2000), is an agenda-driven, bottom-up, unification-based parser. In order to reduce computational demands, state-of-the-art techniques such as subsumption-based packing (Oepen and Carroll, 2000) and the quasi-destructive unification operator (Tomabechi, 1991) have been implemented.

A central component in the parser is the agenda, implemented as a priority queue of parsing tasks (unifications). Tasks are popped from the agenda, until no task is left, after which all passive items spanning the complete sentence are compared with the root conditions as specified by the grammar writer. The best parse is extracted from the parse forest by a Maximum Entropy parse disambiguation model (Toutanova et al., 2002), using selective unpacking (Zhang et al., 2007b).

Two different types of items are identified: passive items and active items. Passive items are 'normal' chart items, in the sense that they can freely combine with other items. Active items still need to combine with a passive item to be complete. Hence, the parser knows two types of tasks as well (see figure 1): *rule+passive* and *active+passive*.

Each time a task succeeds, the following happens:

- For each inserted passive item, add (rule+passive) tasks that combine the passive item with each of the rules, and add (active+passive) tasks that combine with each of the neighbouring active items.
- For each inserted active item, add (active+passive) tasks that combine the remain-

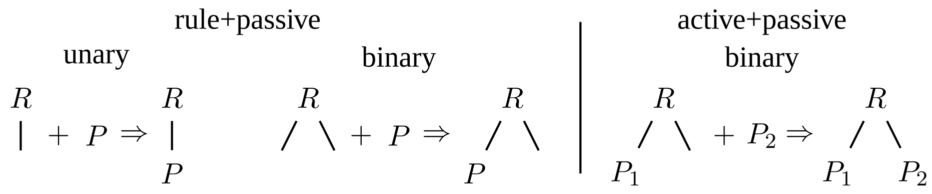


Figure 1: Depicted are the different types of tasks in the PET parser. Not shown are the features structures imposed by the rules and the chart items.

ing gaps in the active item with existing neighbouring passive items in the chart.

2.2 Defining priorities

The priorities of the parsing tasks are calculated based on a generative PCFG model extracted from the treebank by maximum likelihood estimation, smoothed by Lidstone smoothing. Each passive chart item receives a score based on its generative probability, calculated as the product of all applied rule probabilities. For active parsing items, we set the score to be the upper bound of this generative probability, if the item succeeds later in combining with other passive edge(s) to build a complete subtree. This is done by simply assuming the undetermined subtree in the active item receiving a generative score of 1.

The priorities that are assigned to both types of tasks are not yet conditioned on the probability of the topmost rule application. Hence, they are computed using the following simple formula:

$$Pr = p(R) \cdot p(P)$$

where Pr is the task's priority, $p(R)$ the prior probability of the rule category R; and $p(P)$ is the highest possible generative probability of the *resulting* passive item P.

2.3 Restriction strategies

It is a natural thought to allocate more computational resources to longer sentences, and this is exactly what happens in the restriction strategies we develop in this study. We define a cap on the number of tasks for a certain cell/span (i, j), which means that the number of cells is quadratically related to the number of words in a sentence: $n_{cells} = n(n+1)/2$.

We define three task restriction strategies: *all*, *success*, and *passive*. In *all*, the cap is defined for all tasks, whether the unification is successful or not. *Success* only counts tasks that are successful (i.e. lead to either an active or a passive item), and *passive* only counts tasks that lead to a passive item. In all strategies, morphological and lexical tasks are not counted, and hence not restricted. Unary phrasal rules (such as *empty-det*) are counted, though.

The implementation uses only one priority queue. Each time a task is popped from the agenda, it is checked whether the limit for this span has been reached or not. If so, the task is discarded; otherwise, it is executed.

2.4 Methodology

All our experiments are based on the Tiger treebank (Brants et al., 2002). The grammar's lexicon is based on the first 45k sentences in the treebank, and so are the MaxEnt disambiguation model (Toutanova et al., 2002) and the generative model we developed for this study. The development set (s45001-s47500) was used to fine-tune the methods, but all final results presented in this paper are with respect to the test set (s47501-s50000). The maximum time for building up the packed parse forest is 60 seconds, after which unpacking is started. Unpacking the first reading usually has negligible computation costs, and is not reported on. Along with the best reading's derivation, the dependencies are output, and com-

Strategy	exhaustive	all	success	passive
Cap size		3000	200	100
Time (s)	7.20	1.04	0.92	1.06
Coverage	59.4%	60.5%	60.0%	59.0%
Exact	17.6%	17.6%	17.4%	17.4%
Recall	37.6%	39.5%	38.9%	38.0%
Precision	80.7%	80.3%	80.1%	80.4%
F-score	51.3%	52.9%	52.4%	51.6%

Table 1: A more detailed look into some data points from figure 2. 'Coverage' and 'Exact' are sentential percentages, showing how many sentences receive at least one or the exactly correct reading. Recall, precision and f-score are on a per-dependency basis.

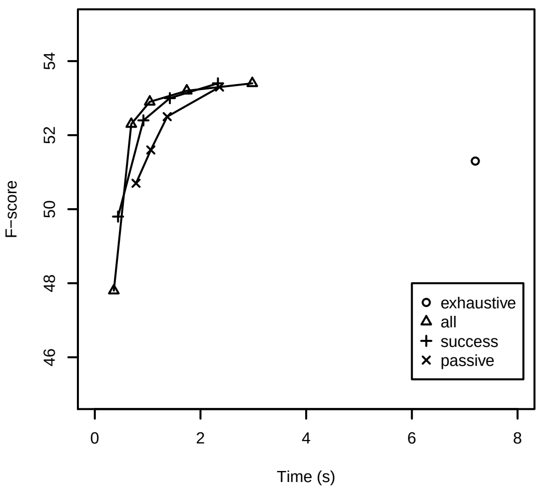


Figure 2: This figure shows the tradeoff between speed and f-score for the standard grammar, using the restriction strategies with different cap sizes.

pared to the gold standard dependencies from the Tiger treebank.

2.5 Results

The results of the experiments, with different cap sizes, are summarized in table 1 and figure 2. As expected, for all strategies it holds that longer computation times lead to higher coverage numbers. The interesting thing is that the restriction of the search space doesn't affect the parses' precision, indicating that the priorities work well: the tasks leading to good solutions are indeed given high priority scores.

A striking observation is that the coverage numbers go up by about 1%, with reductions in parse times of more than 80%. This is due to the use of the timeout, and the generic tendency of our definition of the priorities: because less rule applications lead to higher log probabilities, the agenda will favour tasks with smaller span size. If the agenda doesn't apply too strong a restriction on those tasks, the parser might not create any items spanning the whole sentence after the full 60 seconds, and hence produce no parse. This is mitigated by stronger restriction, leading to a quicker path upwards in the chart.

No large differences of success are found between the different strategies. The intuition behind the *success* and *passive* strategies was that only more effort should be invested into a particular span if not enough chart items for that span have been created. However, the time/quality trade-offs are very similar for all strategies, as shown in figure 2[2].

The strategies we have reported on have one thing in common: their counters are with respect to one particular span, and therefore, they have a very local scope. We have tried other strategies that would give the algorithm more flexibility by defining the caps on more global scale, for instance per span length or for the entire chart. However, this degraded the performance severely, because the parser was not able to divide its attention properly.

[2]One might be tempted to consider the *all* strategy as the best one. However, the time/f-score tradeoff curves look slightly different on the development set.

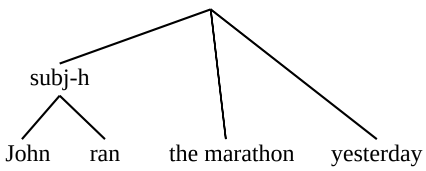
3 Increasing robustness

For hand-written deep parsers, efficiency and coverage are often competing factors: allowing more items to be created might be beneficial for recall, but the parser will also be too slow. However, because the search space can be restricted so rigidly, we can make the grammar more permissive to accept more sentences, hopefully without a heavy efficiency penalty. One way to do this is to remove constraints from the grammar rules. However, that would infringe on the precision-oriented nature of the grammar. Instead, we will keep the normal grammar rules as they are, and create a small number of additional, super-accepting *robustness rules*. The intuition is that when the restricted part of the grammar can find a solution, that solution will indeed be found, and preferred by the statistical models. On the other hand, when the sentence is extragrammatical, the robustness rules may be able to overcome the barriers.

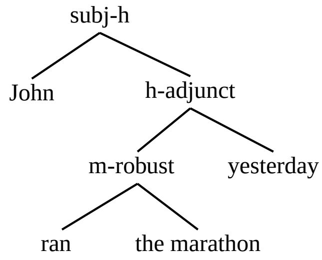
Let's consider the following example, assuming that the grammar only lists 'to run' as an intransitive verb:

'John ran the marathon yesterday'

A fragment approach would come up with the following solution:



'John' will correctly be identified as the subject of 'ran', but that is all. No dependencies are established between 'the marathon' and 'ran', or 'yesterday' and 'ran'. The former is hard to establish, because of the missing lexical item. However, the latter should be doable: the lexicon knows that 'yesterday' is an adverb that modifies verbs. If we could create a robustness rule that would absorb the object ('the marathon') without assigning a dependency, it would at least be able to identify the modifier dependency between 'ran' and 'yesterday'.



In other words, a fragment analysis solely combines items at the top level, whereas a robust parser would ideally be able to overcome barriers in both the lower and the higher regions of the chart, meaning that the damage can be localised and thus minimised. The robustness rules we propose are intended to achieve that.

How does this idea interact with the restriction mechanism explained in the previous section? Robustness rules get an inhibitively large, constant penalty in both the priority model and the disambiguation model. That means that at first the parser will try to build the parse forest with the restricted set of rules, because tasks involving subtrees with only rules from the standard grammar will always have a higher priority than tasks using an item with a robustness rule application in its subtree. When this is finished, the robustness rules try to fill the gaps. Especially in the *success* and *passive* strategies, tasks with robustness rules are discarded if already enough chart items are found for a particular span, meaning that the parser automatically focusses on those parts of the chart that haven't been filled before.

3.1 Defining robustness rules

Defining robustness rules is a sort of grammar engineering, and it took a bit of experimentation to find rules that worked well. One of the factors was the interaction between the subsumption-based packing and the robustness rules. When the chart is built up, items that are subsumed by an existing item are marked as 'frozen', and the latter (more general) item functions as the representative node in the remainder of the parsing process. When unpacking the best solution, the best derivation tree is extracted from the packed forest, which might include a frozen node. Because this frozen node has more constraints than its representative, this derivation tree is not guaranteed to be free of unification failures, and hence, before outputting, this is checked by replaying all the unifications in the derivation tree. This procedure is repeated until a sound derivation has been found.

So what happens when the representative nodes are very general? Many nodes will be packed, and hence the chart will remain compact. However, the unpacking process will become problematic, because many of the proposed derivation trees during unpacking will be incorrect, leading to excessive computation times (in the order of minutes).

Therefore, we chose to define robustness rules such, that the resulting chart items will be equally constrained as their daughters. They are all binary, and have one common ancestor in the type hierarchy:

$$\begin{bmatrix} \textit{structure-robust} & \\ \text{SYNSEM} & \boxed{1} \\ \text{ROBUST} & + \\ \text{MN-DTR} & \begin{bmatrix} \textit{sign} & \\ \text{SYNSEM} & \boxed{1}\begin{bmatrix}\text{LOCAL.CAT.HEAD} & \textit{verb}\end{bmatrix} \\ \text{ROBUST} & - \end{bmatrix} \\ \text{RB-DTR} & \begin{bmatrix} \textit{sign} & \\ \text{SYNSEM} & \begin{bmatrix}\text{NONLOCAL} & \textit{no-nonlocal}\end{bmatrix} \\ \text{ROBUST} & - \end{bmatrix} \end{bmatrix}$$

All rules have a *main* daughter and a *robust* daughter. The co-indexation of the SYNSEM of the main daughter and the SYNSEM of the rule itself has the effect that the resulting chart item will have the exact same syntactic properties as its main daughter, whereas the robust daughter does not contribute to the syntactic properties of the mother node. The ROBUST feature is used to prevent the application of two robust rules consecutively. Additional constraints (not shown) make sure that morphological processing is finished, and that both parts are not involved in a coordination. Robustness rules do not yield a dependency triple (although they mght be guessed accurately by a few heuristics).

We define two pairs of robustness rules, each pair consisting of a rule with MN-DTR first and RB-DTR second, and one rule in the other order:

+V The robust daughter is a verb, which is still allowed to have valence, but cannot have any features in NONLOCAL.

+NV The robust daughter is anything but a verb, cannot have any non-empty valence list, and cannot have any features in NONLOCAL.

3.2 Fragment parsing

As a baseline for comparison, we investigate the existing partial parsing algorithms that pick fragmented analyses from the parse forest as a fallback strategy when there is no full parse available. Kiefer et al. (1999) took a shortest-path approach to find a sequence of fragment analysis that minimizes a heuristics-based cost function. Another variation of the algorithm (Riezler et al., 2001) is to pick fewest chunks that connect the entire sentence. While these early approaches are based on simple heuristics, more sophisticated parse selection methods also use the statistical models to rank the partial analyses. For example, Zhang et al. (2007a) proposed several ways of integrating discriminative parse ranking scores with the partial parse selection algorithm.

In this experiment, we first use the shortest path algorithm to find candidate chunks of partial analysis. All phrasal constituents were given equal weights, and preferred over input and lexical edges. For each chunk (edges spanning the same sub-string of the input sentence), the edge with the highest generative probability is picked. Consequently, the best partial reading (covering that edge) is decoded by the selective unpacking algorithm using the MaxEnt parse ranking model. With each fragment, the partial semantic representations were extracted. Similar to the robustness rules, no cross-fragment dependencies are recovered in this approach. Due to the limited number of chart items and the use of selective unpacking, the computation times for the shortest-path algorithm are marginal.

3.3 Results

The results of this experiment are listed in table 2. For the robust versions of the grammar, no exhaustive parsing results are reported, because they take too long to compute, as can be expected. Coverage number are on a per-sentence

		standard		+V	+NV	+V+NV
		exhaustive	restricted		restricted	
	time (s)	7.20	0.92	4.10	1.42	4.09
no fragment	coverage	59.3%	60.0%	72.6%	69.9%	78.6%
	recall	37.6%	38.9%	48.4%	47.0%	53.8%
	precision	80.7%	80.1%	78.6%	78.2%	77.7%
	f-score	51.3%	52.4%	59.9%	58.7%	63.6%
fragment	coverage	94.3%	98.3%	98.5%	98.7%	98.5%
	recall	50.4%	53.6%	59.5%	56.9%	61.3%
	precision	75.4%	75.0%	75.0%	74.5%	74.7%
	f-score	60.4%	62.5%	66.3%	64.5%	67.3%

Table 2: Results for experiments with different robustness rules, and with or without fragment fallback strategy.

basis, whereas the other percentages are on a per-dependency basis. Time denotes the average number of seconds it takes to build the parse forest. All results under 'restricted' are carried out with the *success* strategy, with a cap of 200 tasks (*success-200*). '(No) fragment' indicates whether a fragment parse is returned when no results are obtained after selective unpacking.

The robustness rules significantly increase the sentential coverage, in the case of +V+NV almost 20 percent points. The gains of +V and +NV are fairly additive: they seem to cover different sets of extragrammatical sentences. In the most permissive setting (+V+NV), dependency recall goes up by 16 percent point, with only a 3 percent point decrease of precision, showing that the newly-covered sentences still receive fairly accurate parses. Also, it can be seen that the +V pair of rules is more effective than +NV to increase coverage. The robust grammars are certainly slower than the standard grammar, but still twice as fast as the standard grammar in an exhaustive setting.

Coverage numbers are approximating 100% when the fragment parsing fallback strategy is applied, in all settings. However, it is interesting to see that the recall numbers are higher when the robustness rules are more permissive, but that no significant effect on the precision is observed. This suggests that the lumps that are connected by the fragment parsing mechanism are larger, due to previous applications of the robustness rules. From this, we conclude that the connections made by the robustness rules are of relatively high quality.

We have also tried the *all*-3000 and *passive*-100 settings (the same as listed in table 1). That yielded very similar results, except on the grammar with both +V and +NV enabled. With *passive*-100, there was a small decrease in coverage (76.0%), but this drop was much more pronounced for *all*-3000: 72.0%. This suggests that, if the pressure on the generative model is larger due to heavier overgeneration, counting successful tasks or passive items performs better than just counting the number of executed tasks.

After manual inspection, we found out that the kind of constructions the robustness rules created were very diverse. Most of the rule applications were not in the top of the tree, as was intended. There also seemed to be a correlation between the length of the robust daughter and the quality of the parse. When the robust daughter of the rule was large, the application of the robustness rule looked like an emergency strategy, with a corresponding quality of the parse. However, when the robustness rule connects a verb to a relatively small constituent (a particle or an NP, for example), the resulting derivation tree was of reasonable quality, keeping most of the other dependencies intact.

4 Discussion

Achieving broad coverage in deep parsing while maintaining high precision is difficult. Until now, most existing hand-written grammar-based parsing systems rely on fragment analyses (or various ways of putting fragments together to compose partial readings), but we argued (with the example in section 3) that such an approach delivers inferior results when the tree falls apart at the very bottom. The use of robust constructions offers a way to keep the damage local, but can create an intractable search space. The proposed pruning strategies carefully control the bound of overgeneration, resulting in improvements on both parsing efficiency and coverage, with a significantly smaller degradation in f-score than a pure fragment approach. The combination of grammar engineering, statistical modelling and algorithmic design in the parser brings the parser performance to a new level.

Although the experiments were carried out on a specific grammar framework, we consider the techniques put forward in this paper to be applicable to other linguistic frameworks. The robustness rules are easy to construct (with the precautions from section 3.1 in mind), and all modern deep parsers have a treebank to their disposal, from which the generative model can be learned.

There are still points that can be improved on. Currently, there is no way to determine which of the *robust* rule applications are more promising than others, and the decision to try one before the other is solely based on the the probabilities of the passive items, and not on the generative model. This can be inefficient: for instance, all robustness rules presented in this paper (both +V and +NV) requires the main daughter to be a verb. It would be straightforward to learn from a small treebank that trying to unify the main daughter of a robustness rules (which should have a verbal head) with a specifier-head rule application does not have a high chance on succeeding.

Another possible improvement is to differentiate between different robustness rules. We presented a two-tier system here, but the framework lends itself naturally to more layers with differing degrees of specificity, creating a smoother scale from specific/prioritised to robust/non-prioritised.

References

Brants, S., S. Dipper, S. Hansen, W. Lezius, and G. Smith. 2002. The TIGER Treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, pages 24–41.

Cahill, A., J.T. Maxwell III, P. Meurer, C. Rohrer, and V. Rosén. 2008. Speeding up LFG parsing using c-structure pruning. In *Proceedings of the Workshop on Grammar Engineering Across Frameworks*, pages 33–40. Association for Computational Linguistics.

Callmeier, U. 2000. PET–a platform for experimentation with efficient HPSG processing techniques. *Natural Language Engineering*, 6(01):99–107.

Clark, S. and J.R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Cramer, B. and Y. Zhang. 2009. Construction of a German HPSG grammar from a detailed treebank. In *Proceedings of the GEAF workshop ACL-IJCNLP 2009*, pages 37–45.

Kiefer, B., H.U. Krieger, J. Carroll, and R. Malouf. 1999. A bag of useful techniques for efficient and robust parsing. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 473–480. Association for Computational Linguistics.

Matsuzaki, T., Y. Miyao, and J. Tsujii. 2007. Efficient HPSG parsing with supertagging and CFG-filtering. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pages 1671–1676, Hyderabad, India.

Ninomiya, T., Y. Tsuruoka, Y. Miyao, and J. Tsujii. 2005. Efficacy of beam thresholding, unification filtering and hybrid parsing in probabilistic HPSG parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 103–114. Association for Computational Linguistics.

Oepen, S. and J. Carroll. 2000. Ambiguity packing in constraint-based parsing: practical results. In *Proceedings of the first conference on North American chapter of the Association for Computational Linguistics*, pages 162–169. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.

Riezler, S., T.H. King, R.M. Kaplan, R. Crouch, J.T. Maxwell III, and M. Johnson. 2001. Parsing the Wall Street Journal using a Lexical-Functional Grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 271–278.

Tomabechi, H. 1991. Quasi-destructive graph unification. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 315–322. Association for Computational Linguistics.

Toutanova, K., C.D. Manning, S. Shieber, D. Flickinger, and S. Oepen. 2002. Parse disambiguation for a rich HPSG grammar. In *Proceedings of the First Workshop on Treebanks and Linguistic Theories*, pages 253–263.

Van Noord, G. 1997. An efficient implementation of the head-corner parser. *Computational Linguistics*, 23(3):425–456.

van Noord, G. 2009. Learning efficient parsing. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 817–825, Athens, Greece, March. Association for Computational Linguistics.

Zhang, Y., V. Kordoni, and E. Fitzgerald. 2007a. Partial parse selection for robust deep processing. In *Proceedings of ACL 2007 Workshop on Deep Linguistic Processing*, pages 128–135, Prague, Czech.

Zhang, Y., S. Oepen, and J. Carroll. 2007b. Efficiency in Unification-Based N-Best Parsing. In *Proceedings of the Tenth International Conference on Parsing Technologies*, pages 48–59. Association for Computational Linguistics.

Local lexical adaptation in Machine Translation through triangulation: SMT helping SMT

Josep Maria Crego
LIMSI-CNRS
jmcrego@limsi.fr

Aurélien Max
LIMSI-CNRS
Univ. Paris Sud
amax@limsi.fr

François Yvon
LIMSI-CNRS
Univ. Paris Sud
yvon@limsi.fr

Abstract

We present a framework where auxiliary MT systems are used to provide lexical predictions to a main SMT system. In this work, predictions are obtained by means of pivoting via auxiliary languages, and introduced into the main SMT system in the form of a low order language model, which is estimated on a sentence-by-sentence basis. The linear combination of models implemented by the decoder is thus extended with this additional language model. Experiments are carried out over three different translation tasks using the European Parliament corpus. For each task, nine additional languages are used as auxiliary languages to obtain the triangulated predictions. Translation accuracy results show that improvements in translation quality are obtained, even for large data conditions.

1 Introduction

Important improvements are yet to come regarding the performance of Statistical Machine Translation systems. Dependence on training data and limited modelling expressiveness are the focus of many research efforts, such as using monolingual corpora for the former and syntactic models for the latter.

Another promising approach consists in exploiting complementary sources of information in order to build better translations, as done by consensus-based system combination (e.g. (Matusov et al., 2008)). This, however, requires to have several systems available for the same language pair. Considering that the same training data would be available to all systems, differences in translation modelling are expected to produce redundant and complementary hypotheses. Multisource translation (e.g. (Och and Ney, 2001; Schwartz, 2008)) is a variant, involving source texts available in several languages which can be translated by systems for different language pairs and whose outputs can be successfully combined into better translations (Schroeder et al., 2009). One theoretical expectation of multisource translation is that it can successfully reduce ambiguity of the original source text, but does so under the rare conditions of availability of existing (accurate) translations. In contrast, pivot-based system combination (e.g. (Utiyama and Isahara, 2007; Wu and Wang, 2007)) aims at compensating the lack of training data for a given language pair by producing translation hypotheses obtained by pivoting via an intermediary language for which better systems are available.

These techniques generally produce a search space that differs from that of the direct translation systems. As such, they create a new translation system out of various systems for which diagnosis becomes more difficult.

This paper instead focusses on improving a single system, which should be state-of-the-art as regards data and models. We propose a framework in which information coming from external sources is used to boost lexical choices and guide the decoder into making more informed choices.[1]

[1] We performed initial experiments where the complementary information was exploited during *n*-best list reranking (Max et al., 2010), but except for the multisource condition the list of hypotheses contained too little useful variation

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 232–240,
Beijing, August 2010

Complementary sources can be of different nature: they can involve other automatic systems (for the same or different language pairs) and/or human knowledge. Furthermore, complementary information is injected at the lexical level, thus making targeted fine-grained lexical predictions useful. Importantly, those predictions are exploited at the sentence level[2], so as to allow for efficient use of source contextual information.

The second contribution of this paper is an instantiation of the proposed framework. Automatically pivoting via auxiliary languages is used to make complementary predictions that are exploited through language model adaptation by the decoder for a given language pair. For this apparently difficult condition, where predictions result from automatic translations involving two systems, we manage to report significant improvements, measured with respect to the target and the source text, under various configurations.

This paper is organized as follows. We first review related work in section 2.1, and describe the distinctive characteristics of our approach in Section 2.2. Section 2.3 presents our instantiation of the framework based on lexical boosting via auxiliary language triangulation. Experiments involving three language pairs of various complexity and different amounts of training data are described in Section 3. We finally conclude by discussing the prospects offered by our proposed framework in Section 4.

2 A framework for sentence-level lexical boosting

2.1 Related work

The idea of using more than one translation system to improve translation performance is not new and has been implemented in many different ways which we briefly review here.

System combination An often used strategy consists in *combining the output of several systems* for a fixed language pair, and to rescore the resulting set of hypotheses taking into account all the available translations and scores. Various proposals have been made to efficiently perform such a combination, using auxiliary data structures such as n-best lists, word lattices or consensus networks (see for instance (Kumar and Byrne, 2004; Rosti et al., 2007; Matusov et al., 2008; Hildebrand and Vogel, 2008; Tromble et al., 2008)). Theses techniques have proven extremely effective and have allowed to deliver very significant gains in several recent evaluation campaigns (Callison-Burch et al., 2008).

Multisource translation A related, yet more resourceful approach, consists in trying to combine several systems providing translations *from different sources into the same target*, provided such *multilingual sources* are available. (Och and Ney, 2001) propose to select the most promising translation amongst the hypotheses produced by several Foreign→English systems, where output selection is based on the translation scores. The intuition that if a system assigns a high figure of merits to the translation of a particular sentence, then this translation should be preferred, is implemented in the MAX combination heuristics, whose relative (lack of) success is discussed in (Schwartz, 2008). A similar idea is explored in (Nomoto, 2004), where the sole target language model score is used to rank competing outputs. (Schroeder et al., 2009) propose to combine the available sources prior to translation, under the form of a multilingual lattice, which is decoded with a multisource phrase table. (Chen et al., 2008) integrate the available auxiliary information in a different manner, and discuss how to improve the translation model of the primary system: the idea is to use the entries in the phrase table of the auxiliary system to filter out those accidental correspondences that pollute the main translation model. The most effective implementation of multisource translation to date however consists in using mono-source system combination techniques (Schroeder et al., 2009).

Translation through pivoting The use of auxiliary systems has also been proposed in another common situation, as a possible remedy to the lack of parallel data for a particular language pair, or for a particular domain. Assume, for instance, that one wishes to build a translation system for

to lead to measurable improvements.

[2]We plan to experiment next on using predictions at the document level.

the pair $A \rightarrow B$, for which the parallel data is sparse; assuming further that such parallel resources exist for pairs $A \rightarrow C$ and for $C \rightarrow B$, it is then tempting to perform the translation indirectly through *pivoting*, by first translating from A to C, then from C to B. Direct implementations of this idea are discussed e.g. in (Utiyama and Isahara, 2007). Pivoting can also intervene earlier in the process, for instance as a means to *automatically generate* the missing parallel resource, an idea that has also been considered to adapt an existing translation systems to new domains (Bertoldi and Federico, 2009). Pivoting can finally be used to fix or improve the translation model: (Cohn and Lapata, 2007) augments the phrase table for a baseline bilingual system with supplementary phrases obtained by pivoting into a third language.

Triangulation in translation Triangulation techniques are somewhat more general and only require the availabily of *one* auxiliary system (or one auxiliary parallel corpus). For instance, the authors of (Chen et al., 2008) propose to use the translation model of an auxiliary $C \rightarrow B$ system to filter-out the phrase-table of a primary $A \rightarrow B$ system.

2.2 Our framework

As in other works, we propose to make use of several MT systems (of any type) to improve translation performance, but contrarily to these works we concentrate on *improving one particular system*. Our framework is illustrated on Figure 1. The main system (henceforth, *direct* system), corresponding to configuration **1**, is a SMT system, translating from German to English in the example. Auxiliary information may originate from various sources (**2**-**6**) and enter into the decoder. A new model is dynamically built and is used to guide the exploration of the search space to the best hypothesis. Several auxiliary models can be used at once and can be weighted by standard optimization techniques using development data, so that bad sources are not used in practice, or by exploiting *a priori* information. In the implementation described in section 2.3, this information is updated by the auxiliary source at each sentence.

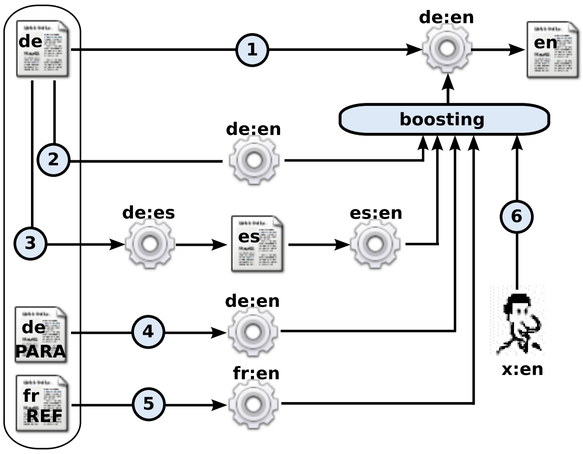


Figure 1: Lexical boosting framework with various configurations for auxiliary predictions

We now briefly describe various possible configurations to make some links to previous works explicit. Configuration **2** translates the same source text by means of another system for the same language pair, as would be done in system combination, except that here a new complete decoding is performed by the direct system. Configuration **3**, which will be detailed in section 2.3, uses translations obtained by triangulating via an auxiliary language (Spanish in the example). Using this two-step translation is common to pivot approaches, but our approach is different in that the result of the triangulation is only used as auxiliary information for the decoding of the direct system. Configurations **4** and **5** are instances of multisource translation, where a paraphrase or a translation of the source text is available. Lastly, configuration **6** illustrates the case where a human translator, with knowledge of the target language and at least of one of the available source languages, could influence the decoding by providing *desired*[3] words (e.g. only for source words or phrases that would be judged difficult to translate). This human supervision through a feedback text in real time is similar to the proposal of (Dymetman et al., 2003).

Given this framework, several questions arise,

[3]The proposal as it is limits the hypotheses produced by the system to those that are attainable given its training data. It is conceivable, however, to find ways of introducing new knowledge in this framework.

the most important underlying this work being whether the performance of SMT systems can be improved by using other SMT systems. Another point of interest is whether improvements made to *auxiliary* systems can yield improvement to the *direct* system, without the latter undergoing any modification.

2.3 Lexical boosting via triangulation

Auxiliary translations obtained by pivoting can be viewed as a source of adaptation data for the target language model of the direct system. Assuming we have computed n-best translation hypotheses of a sentence in the target language, we can then boost the likeliness of the words and phrases occurring in these hypotheses by deriving an auxiliary language model for each test sentence. This allows us to integrate this auxiliary information during the search and thus provides a tighter integration with the direct system. This idea has successfully been used in speech recognition, using for instance close captions (Placeway and Lafferty, 1996) or an imperfect translation (Paulik et al., 2005) to provide auxiliary in-domain adaptation data for the recognizer's language model. (Simard and Isabelle, 2009) proposed a similar approach in Machine Translation in which they use the target-side of an exact match in a translation memory to build language models on a per sentence basis used in their decoder.

This strategy can be implemented in a straightforward manner, by simply training a language model using the n-best list as an adaptation corpus. Being automatically generated, hypotheses in the n-best list are not entirely reliable: in particular, they may contain very unlikely target sequences at the junction of two segments. It is however straightforward to filter these out using the available phrase alignment information.

This configuration is illustrated on Figure 2: the direct system (configuration **1**) makes use of predictions from pivoting through an auxiliary language (configuration **2**), where n-best lists can be used to produce several hypotheses. In order to get a upper bound on the potential gains of this approach, we can run the artificial experiment (configuration **3**) where a reference in the target language is used as a "perfect" source of information. Furthermore, we are interested in the performance of the simple pivot system alone (configuration **4**), as it gives an indication of the quality of the data used for LM adaptation.

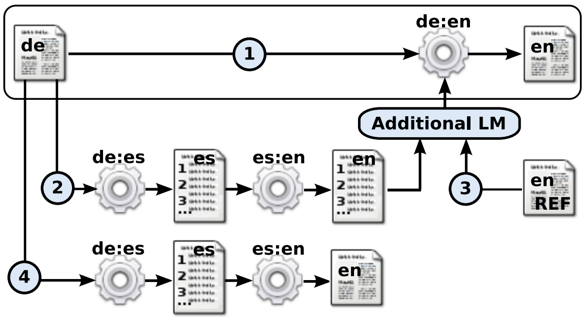


Figure 2: Architecture of a German→English system for lexical boosting via triangulation through Spanish

3 Experiments and results

3.1 Translation engine

In this study, we used our own machine translation engine, which implements the n-gram-based approach to statistical machine translation (Mariño et al., 2006). The translation model is implemented as a stochastic finite-state transducer trained using a n-gram language model of (source,target) pairs.

In addition to a bilingual n-gram model, our SMT system uses six additional models which are linearly combined following a discriminative modeling framework: two *lexicalized reordering* (Tillmann, 2004) models,a *target-language model*, two *lexicon models*, a 'weak' distance-based *distortion model*, a *word bonus model* and a *translation unit bonus model*. Coefficients in this linear combination are tuned over development data with the MERT optimization toolkit[4], slightly modified to use our decoder's n-best lists.

For this study, we used 3-gram bilingual and 3-gram target language models built using modified Kneser-Ney smoothing (Chen and Goodman, 1996); model estimation was performed with the SRI language modeling toolkit.[5] Target language

[4] `http://www.statmt.org/moses`

[5] `http://wwww.speech.sri.com/projects/srilm`

models were trained on the target side of the bitext corpora.

After preprocessing the corpora with standard tokenization tools, word-to-word alignments are performed in both directions, source-to-target and target-to-source. In our system implementation, the *GIZA++* toolkit[6] is used to compute the word alignments. Then, the *grow-diag-final-and* heuristic is used to obtain the final alignments from which translation units are extracted. Convergent studies have showed that systems built according to these principles typically achieve a performance comparable to that of the widely used MOSES phrase-based system for the language pairs under study.

3.2 Corpora

We have used the Europarl corpus[7] for our main and auxiliary languages. The eleven languages are: Danish (da), German (de), English (en), Spanish (es), Finnish (fi), French (fr), Greek (el), Italian (it), Dutch (nl), Portuguese (pt) and Swedish (sv).

We focussed on three translation tasks: one for which translation accuracy, as measured by automatic metrics, is rather high ($fr \rightarrow en$), and two for which translation accuracy is lower ($de \rightarrow en$) and ($fr \rightarrow de$). This will allow us to check whether the improvements provided by our method carry over even in situations where the baseline is strong; conversely, it will allow us to assess whether the proposed techniques are applicable when the baseline is average or poor.

In order to measure the contribution of each of the auxiliary languages we used a subset of the training corpus that is common to all language pairs, hereinafter referred to as the *intersection* data condition. We used the English side of all training language pairs to collect the same sentences in all languages, summing up to $320,304$ sentence pairs. Some statistics on the data used in this study are reported in Table 1. Finally, in order to assess the impact of the training data size over the results obtained, we also considered a much more challenging condition for the $fr \rightarrow de$ pair, where we used the entire Europarl data (V5) made available for the fifth Workshop on Statistical Machine Translation[8] for training, and test our system on out-of-domain news data. The training corpus in this condition contains 43.6M French words and 37.2M German words.

Development and test data for the first condition (*intersection*) were obtained by leaving out respectively 500 and 1000 sentences from the common subset (same sentences for all languages), while the first 500 sentences of *newstest2008* and the entire *newstest2009* official test sets were used for the *full* data condition.

	Train		*Dev*			*Test*		
	Words	*Voc.*	*Words*	*Voc.*	*OOV*	*Words*	*Voc.*	*OOV*
da	$8.5M$	$133.5k$	$13.4k$	$3.2k$	104	$25.9k$	$5.1k$	226
de	$8.5M$	$145.3k$	$13.5k$	$3.5k$	120	$26.0k$	$5.5k$	245
en	$8.9M$	$53.7k$	$14.0k$	$2.8k$	39	$27.2k$	$4.0k$	63
es	$9.3M$	$85.3k$	$14.6k$	$3.3k$	56	$28.6k$	$5.0k$	88
fi	$6.4M$	$274.9k$	$10.1k$	$4.3k$	244	$19.6k$	$7.1k$	407
fr	$10.3M$	$67.8k$	$16.1k$	$3.2k$	47	$31.5k$	$4.8k$	87
el	$8.9M$	$128.3k$	$14.1k$	$3.9k$	72	$27.2k$	$6.2k$	159
it	$9.0M$	$78.9k$	$14.3k$	$3.4k$	61	$28.1k$	$5.1k$	99
nl	$8.9M$	$105.0k$	$14.2k$	$3.1k$	76	$27.5k$	$4.8k$	162
pt	$9.2M$	$87.3k$	$14.5k$	$3.4k$	49	$28.3k$	$5.2k$	118
sv	$8.0M$	$140.8k$	$12.7k$	$3.3k$	116	$24.5k$	$5.2k$	226

Table 1: *Statistics for the training, development and test sets of the intersection data condition*

3.3 Results

In this section, we report on the experiments carried out to assess the benefits of introducing an auxiliary language model to the linear combination of models implemented in our SMT system.

Table 2 reports translation accuracy (BLEU) results for the main translation tasks considered in this work ($fr \rightarrow de$), ($fr \rightarrow en$) and ($de \rightarrow en$), as well as for multiple intermediate tasks needed for pivoting via auxiliary systems.

For each triplet of languages (*src*, *aux*, *trg*), columns 4^{th} to 6^{th} show BLEU scores for systems performing ($src \rightarrow aux$), ($aux \rightarrow trg$) and *pivot* translations using *aux* as the bridge language.

The last two columns display BLEU scores for the main translation tasks ($fr \rightarrow de$), ($fr \rightarrow en$) and ($de \rightarrow en$). Column *src-trg* refers to the baseline (direct) systems, for which no additional lan-

[6] http://www.fjoch.com/GIZA++.html

[7] http://www.statmt.org/europarl

[8] http://www.statmt.org/wmt10

src aux trg	*src-aux*	*aux-trg*	*pivot*	*src-trg*	*+auxLM*
Intersection data condition					
fr - de	-	-	-	18.02	
da	22.78	20.02	16.27		+0.44
el	24.54	18.51	15.86		+0.76
en	29.53	17.31	15.69		+0.50
es	**34.94**	18.31	**16.76**		**+0.96**
fi	10.71	14.15	11.39		+0.65
it	31.60	16.86	16.54		-0.05
nl	22.71	**21.44**	**16.76**		+0.55
pt	33.61	17.47	16.34		-0.12
sv	20.73	19.59	13.73		-0.14
average					*+0.39*
- - ref	-	-	-	-	+6.46
fr - en	-	-	-	29.53	
da	22.78	29.54	25.48		+0.02
de	18.02	24.66	23.50		+0.05
el	24.54	29.37	25.31		+0.07
es	**34.94**	**31.05**	**27.76**		**+0.61**
fi	10.71	20.56	19.15		+0.44
it	31.60	25.75	25.79		+0.32
nl	22.71	24.49	25.15		+0.01
pt	33.61	29.44	27.27		+0.01
sv	20.73	30.98	23.74		+0.50
average					*+0.22*
- - ref	-	-	-	-	+11.30
de - en	-	-	-	24.66	
da	24.59	29.54	**22.73**		**+0.96**
el	19.72	29.37	20.88		**+1.02**
es	**25.48**	**31.05**	21.23		+0.77
fi	12.42	20.56	18.02		**+0.94**
fr	25.93	29.53	21.55		+0.19
it	18.82	25.75	18.05		+0.19
nl	24.97	24.49	22.62		+0.64
pt	23.15	29.44	21.93		+0.87
sv	19.80	**30.98**	21.35		+0.69
average					*+0.69*
- - ref	-	-	-	-	+9.53
Full data condition					
fr - de	-	-	-	19.94	
es	38.76	20.18	19.36		+0.61

Table 2: *Translation accuracy (BLEU) results.*

guage model is used; column *+auxLM* refers to the same system augmented with the additional language model. Additional language models are built from hypotheses obtained by means of *pivot* translations, using *aux* as auxiliary language. The last score is shown in the form of the difference (improvement) with respect to the score of the baseline system.

This table additionally displays the BLEU results obtained when building the additional language models directly from the English reference translations (see last row of each translation task). These numbers provide an upper-bound of the expected improvements. Note finally that numbers in boldface correspond to the best numbers in their column for a given language pair.

As detailed above, the additional language models are built using *trg* hypotheses obtained by pivoting via an auxiliary language: $(src \rightarrow aux) + (aux \rightarrow trg)$. Hence, column *pivot* shows the quality (measured in terms of BLEU) of the hypotheses used to estimate the additional model. Note that we did not limit the language model to be estimated from the 1-best *pivot* hypotheses. Instead, we uses n-best translation hypotheses of the $(src \rightarrow aux)$ system and m-best hypotheses of the $(aux \rightarrow trg)$ system. Hence, $n \times m$ target hypotheses were used as training data to estimate the additional models. Column *+auxLM* shows BLEU scores over the test set after performing four system optimizations on the development set to select the best combination of values used for n and m among: $(1, 1)$, $(10, 1)$, $(10, 1)$ and $(10, 10)$. All hypotheses used to estimate a language model are considered equally likely. Language models are learnt using Witten-Bell discounting. Approximately ± 1.0 point must be added to BLEU scores shown in the last 2 columns for 95% confidence levels.

As expected, *pivot* translations yield lower quality scores than the corresponding direct translations hypotheses. However, *pivot* hypotheses may contain better lexical predictions, that the additional model helps transfer into the baseline system, yielding translations with a higher quality, as shown in many cases the *+auxLM* systems results. The case of using Finnish as an auxiliary language is particularly remarkable. Even though *pivot* hypotheses obtained through Finnish have the lowest scores[9], they help improve the baseline performance as additional language models.

As expected, the translation results of the pair

[9] Given the agglutinative nature of morphological processes in Finnish, reflected in a much lower number of words per sentence, and a higher number of types (see Table 1), BLEU scores for this language do not compare directly with the ones obtained for other languages.

with a highest baseline ($fr \rightarrow en$) were on average less improved than those of the pairs with lower baselines.

As can also be seen, the contribution of each auxiliary language varies for each of the three translation tasks. For instance, Danish (da) provides a clear improvement to ($de \rightarrow en$) translations, while no gain is observed for ($fr \rightarrow en$). No clear patterns seems to emerge, though, and the correlation between the quality of the pivot translation and the boost provided by using these pivot hypotheses remains to be better analyzed.

In order to assess whether the improvements obtained carry over larger data conditions, we trained our ($fr \rightarrow de$), ($fr \rightarrow es$) and ($es \rightarrow de$) systems over the entire EPPS data. Results are reported in the bottom part of Table 2. As can be seen, the ($fr \rightarrow de$) system is still improved by using the additional language model. However, the absolute value of the gain under the *full* condition ($+0.61$) is lower than that of the *intersection* data condition ($+0.96$).

3.4 Contrastive evaluation of lexical translation

In some cases, automatic metrics such as BLEU cannot show significant differences that can be revealed by fine-grained focussed human evaluation (e.g. (Vilar et al., 2006)). Furthermore, computing some similarity between a system's hypotheses and *gold standard* references puts a strong focus on the target side of translation, and does not allow evaluating translation performance from the source words that were actually translated. We therefore use the evaluation methodology described in (Max et al., 2010) for a complementary measure of translation performance that focuses on the contrastive ability of two systems to adequately translate source words.

Source words from the test corpus were first aligned with target words in the reference, by automatically aligning the union of the training and test corpus using GIZA++.[10] The test corpus was analyzed by the TREETAGGER[11] so as to identify content words, which have a more direct impact on translation adequacy. When source words are aligned to several target words, each target word should be individually searched for in the candidate translation, and words from the reference can only be matched once.

[10]The obtained alignments are thus strongly influenced by alignments from the training corpus. It could be noted that alignments could be manually corrected.

[11]`http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger`

		Source words' part-of-speech						
aux		**ADJ**	**ADV**	**NOM**	**PRO**	**VER**	all	**+Bleu**
el	-	27	21	114	25	99	286	+0.07
	+	62	29	136	27	114	368	
es	-	33	25	106	26	110	300	+0.61
	+	64	38	136	22	117	377	
fi	-	44	40	106	20	92	302	+0.44
	+	49	31	120	23	106	329	
it	-	55	39	128	35	119	376	+0.32
	+	55	39	145	36	121	396	
sv	-	40	30	138	29	109	346	+0.50
	+	69	46	144	23	134	416	

Table 3: Contrastive lexical evaluation results per part-of-speech between the baseline French→English system and our systems using various auxiliary languages. '-' (resp. '+') values indicate numbers of words that only the baseline system (resp. our system) correctly translated with respect to the reference translation.

Table 3 shows contrastive results per part-of-speech between the baseline fr→en system and systems using various auxiliary languages. Values in the '-' row indicate the number of words that only the baseline system translated as in the reference translation, and values in the '+' row the number of words that only our corresponding system translated as in the reference. The most striking result is the contribution of Greek, which, while giving no gain in terms of BLEU, improved the translation of 82 content words. This could be explained, in addition to the lower Bleu3 and Bleu4 precision, by the fact that the quality of the translation of grammatical words may have decreased. On the contrary, Italian brings little improvement for content words save for nouns. The mostly negative results on the translation of pronouns were expected, because this depends on their antecedent in English and is not the object of specific modelling from the systems. The translation of nouns and adjectives benefits the most from auxiliary translations.

Figure 3 illustrates this evaluation by means of two examples. It should be noted that a recurrent type of improvement was that of avoiding missing words, which is here a direct result of their being boosted in the auxiliary hypotheses.

4 Conclusions and future work

We have presented a framework where auxiliary MT systems are used to provide useful information to a main SMT system. Our experiments on auxiliary language triangulation have demonstrated its validity on a difficult configuration and have shown that improvements in translation quality could be obtained even under large training data conditions.

The fact that low quality sources such as pivot translation can provide useful complementary information calls for a better understanding of the phenomena at play. It is very likely that, looking at our results on the contribution of auxiliary languages, improving the quality of an auxiliary source can also be achieved by identifying what a source is good for. For example, in the studied language configurations predictions of translations for pronouns in the source text by auxiliary triangulation does not give access to useful information. On the contrary, triangulation with Greek when translating from French to English seems to give useful information regarding the translation of adjectives, a result which was quite unexpected.

Also, it would be interesting to use richer predictions than short *n*-grams, such as syntactic dependencies, but this would require significant changes on the decoders used. Using dynamic models at the discourse level rather than only at the sentence level would also be a useful improvement. Besides the improvements just mentioned, our future work includes working on several configurations of the framework described in section 2.2, in particular investigating the new type of system combination.

Acknowledgements

This work has been partially funded by OSEO under the Quaero program.

References

Bertoldi, Nicola and Marcello Federico. 2009. Domain adaptation for statistical machine translation with monolingual resources. In *Proceedings of WMT*, Athens, Greece.

Callison-Burch, Chris, Cameron Shaw Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2008. Further meta-evaluation of machine translation. In *Proceedings of WMT*, Columbus, USA.

Chen, Stanley F. and Joshua T. Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proceedings of ACL*, Santa Cruz, USA.

Chen, Yu, Andreas Eisele, and Martin Kay. 2008. Improving statistical machine translation efficiency by triangulation. In *Proceedings of LREC*, Marrakech, Morocco.

Cohn, Trevor and Mirella Lapata. 2007. Machine translation by triangulation: Making effective use of multi-parallel corpora. In *Proceedings of ACL*, Prague, Czech Republic.

Dymetman, Marc, Aurélien Max, and Kenji Yamada. 2003. Towards interactive text understanding. In *Proceedings of ACL, short paper session*, Sapporo, Japan.

Hildebrand, Almut Silja and Stephan Vogel. 2008. Combination of machine translation systems via hypothesis selection from combined n-best lists. In *Proceedings of AMTA*, Honolulu, USA.

Kumar, Shankar and William Byrne. 2004. Minimum bayes-risk decoding for statistical machine translation. In *Proceedings of NAACL-HLT*, Boston, USA.

Mariño, José, Rafael E. Banchs, Josep Maria Crego, Adria de Gispert, Patrick Lambert, J.A.R. Fonollosa, and Martha Costa-jussà. 2006. N-gram based machine translation. *Computational Linguistics*, 32(4):527–549.

Matusov, Evgeny, Gregor Leusch, Rafael E. Banchs, Nicola Bertoldi, Daniel Dechelotte, Marcello Federico, Muntsin Kolss, Young-Suk Lee, Jose Mariño, Matthias Paulik, Salim Roukos, Holger Schwenk, and Hermann Ney. 2008. System combination for machine translation of spoken and written language. *IEEE Transactions on Audio, Speech and Language Processing*, 16(7):1222–1237, September.

Max, Aurélien, Josep M. Crego, and François Yvon. 2010. Contrastive Lexical Evaluation of Machine Translation. In *Proceedings of LREC*, Valletta, Malta.

ref #357	this concession to the unions ignores the reality that all airlines have different safety procedures which even differ between aircrafts within each airline .
bas	this **concession unions ignores** the *fact* that all **airlines have different safety procedures** which are **even within each** of the *companies* in accordance with the types of equipment .
w.r.t. src	cette **concession** aux **syndicats ignore** la *réalité* selon laquelle toutes les **compagnies aériennes ont** des **procédures de sécurité différentes** qui diffèrent **même** au **sein** de chacune des *compagnies* en fonction des types d ' *appareils* .
+aux	this **concession** to the trade **unions ignores** the **reality** according to which all the **airlines have different safety procedures** which **differ even within each** of the *companies* in accordance with the types of equipment .
w.r.t. src	cette **concession** aux **syndicats ignore** la **réalité** selon laquelle toutes les **compagnies aériennes ont** des **procédures de sécurité différentes** qui **diffèrent même** au **sein** de chacune des *compagnies* en fonction des types d ' *appareils* .

Figure 3: Example of automatic translations from French to English for the baseline system and when using Spanish as the auxiliary language. Bold marking indicates source/target words which were correctly translated according to the reference translation.

Nomoto, Tadashi. 2004. Multi-engine machine translation with voted language model. In *Proceedings of ACL*, Barcelona, Catalunya, Spain.

Och, Franz Josef and Hermann Ney. 2001. Statistical multi-source translation. In *Proceedings of MT Summit*, Santiago de Compostela, Spain.

Paulik, Matthias, Christian Fügen, Thomas Schaaf, Tanja Schultz, Sebastian Stüker, and Alex Waibel. 2005. Document driven machine translation enhanced automatic speech recognition. In *Proceedings of InterSpeech*, Lisbon, Portugal.

Placeway, Paul and John Lafferty. 1996. Cheating with imperfect transcripts. In *Proceedings of ICSLP*, Philadelphia, USA.

Rosti, Antti-Veikko, Necip Fazil Ayan, Bin Xiang, Spyros Matsoukas, Richard Schwatz, and Bonnie J. Dorr. 2007. Combining outputs from multiple machine translation systems. In *Proceedings of NAACL-HTL*, Rochester, USA.

Schroeder, Josh, Trevor Cohn, and Philipp Koehn. 2009. Word lattices for multi-source translation. In *Proceedings of EACL*, Athens, Greece.

Schwartz, Lane. 2008. Multi-source translation methods. In *Proceedings of AMTA*, Honolulu, USA.

Simard, Michel and Pierre Isabelle. 2009. Phrase-based machine translation in a computer-assisted translation environment. In *Proceedings of Machine Translation Summit XII*, Ottawa, Canada.

Tillmann, Christoph. 2004. A unigram orientation model for statistical machine translation. In *Proceedings of NAACL-HLT*, Boston, USA.

Tromble, Roy, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. Lattice Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of EMNLP*, Honolulu, USA.

Utiyama, Masao and Hitoshi Isahara. 2007. A comparison of pivot methods for phrase-based statistical machine translation. In *Proceedings of NAACL-HLT*, Rochester, USA.

Vilar, David, Jia Xu, Luis Fernando d'Haro, and Hermann Ney. 2006. Error Analysis of Statistical Machine Translation Output. In *Proceedings of LREC*, Genoa, Italy.

Wu, Hua and Haifeng Wang. 2007. Pivot language approach for phrase-based statistical machine translation. In *Proceedings of ACL*, Prague, Czech Republic.

Automated Translation of Semantic Relationships

Dmitry Davidov
ICNC
Hebrew University of Jerusalem

Ari Rappoport
Institute of Computer Science
Hebrew University of Jerusalem
arir@cs.huji.ac.il

Abstract

We present a method for translating semantic relationships between languages where relationships are defined as pattern clusters. Given a pattern set which represents a semantic relationship, we use the web to extract sample term pairs of this relationship. We automatically translate the obtained term pairs using multilingual dictionaries and disambiguate the translated pairs using web counts. Finally we discover the set of most relevant target language patterns for the given relationship. The obtained pattern set can be utilized for extraction of new relationship examples for the target language.

We evaluate our method on 11 diverse target languages. To assess the quality of the discovered relationships, we use an automatically generated cross-lingual SAT analogy test, WordNet relationships, and concept-specific relationships, achieving high precision. The proposed framework allows fully automated cross-lingual relationship mining and construction of multilingual pattern dictionaries without relying on parallel corpora.

1 Introduction

Acquiring and understanding semantic relationships is crucial for many NLP applications. In many cases, we would like to know if a given term pair participates in a specified semantic relationship or if two different term pairs encode the same (possibly unspecified) type of relationship. Beyond the well-known major relationship types such as hyponymy (is-a) and meronymy (part-of), there is a huge number of other relationships between objects and concepts. Examples include general relations such as larger-than, contained-in, liked-by and domain specific ones such as country-language, product-manufacturer, product-seller, drug-disease etc.

The vast majority of NLP research is done in a few languages for which extensive corpora (including the web) are available. As a result, most relationship retrieval studies and lexical database compilation efforts target only a few languages. However, due to the substantial growth of the multilingual web[1] and a growing demand for NLP application coverage for less common languages, there is a need for relationship data in many less studied languages.

In this paper we address the task of translating relationships between languages, which has two obvious benefits. First, it can directly help applications such as machine translation, cross-lingual information retrieval, cross-lingual web mining and the construction and enrichment of semantic databases. Second, it can assist applications in a single language, especially when compensating for a relative scarcity of resources in that language. We focus on relations between two entities, which are the most common type.

When discussing the translation of relationships, it is important to define how these are represented and in what way the task differs from MT. While there is wide agreement on the definition and representation of major relationship types such as hypernymy and (to a lesser extent) meronymy, there is no single accepted method (or

[1] http://www.internetworldstats.com/stats7.htm

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 241–249,
Beijing, August 2010

resources) for other less common relationships. Among the methods that have been proposed for specifying lexical relationships are natural language description and rules (Girju et al., 2007), distributional means (Turney, 2005), sample term pairs (Pasca et al, 2006), relationship instances (Banko et al., 2007) and pattern clusters (Davidov and Rappoport, 2008a).

In this paper we utilize the last definition. Following (Davidov and Rappoport, 2008a) each semantic relationship can be defined and represented by a set of lexical patterns such that the represented relation holds between entities filling the patterns' slots. We focus on pattern clusters relationship definition due to several reasons. First, as opposed to natural language descriptions, pattern clusters are formal. Second, as opposed to the other methods above, pattern clusters provide a 'generative' model for the represented relationship – it is possible to obtain from them relationship instances and term pairs, as we indeed utilize in this paper. Third, pattern clusters can be mined in a fully unsupervised manner, or in a focused manner when the relationship desired is known. Finally, pattern methods have proven to be highly efficient and effective for lexical acquisition tasks (Pantel et al, 2004; Davidov and Rappoport, 2006).

The proposed framework comprises the following stages. First, given a set of patterns defining a relationship in a source language, we obtain from the web a set of corresponding term pairs. Next, for each of the terms in the obtained term pairs, we retrieve sets of their translations to the target language using available multilingual dictionaries. Now that we have a set of translations for each term in each pair, we retrieve search engine snippets with the translated term pairs. We then select appropriate word senses using web counts, and extract a set of patterns which connect these disambiguated terms. As a result we get a set of relation-specific target language patterns, effectively obtaining the desired relationship definition. We can optionally use the retrieved pattern sets to obtain term pairs of target language relationships from the web.

We performed a thorough evaluation for various relationships involving 11 languages. We tested our framework on major relationships like meronymy, specific relationships like country-capital and unspecified unsupervisedly discovered English relationships. The obtained relationships were manually verified by human judges using cross-lingual SAT analogy questions, and a few specific factual relationships were evaluated using a gold standard.

Our main contribution is a novel framework for automated relationship translation across languages, where relationships are defined as pattern clusters or as term pairs. This framework allows fully automated cross-lingual relationship mining and construction of multilingual pattern dictionaries without relying on parallel corpora.

In Section 2 we discuss related work. Section 3 details the algorithm. Section 4 describes the evaluation, and Section 5 concludes.

2 Related work

Recently, with the development of practical applications which utilize WN-like databases in dozens of languages, great effort has been made to manually construct and interconnect such databases for different languages (Pease et al, 2008; Charoenporn et al., 2007). Some studies (e.g., (Amasyali, 2005)) use semi-automated methods based on language-specific heuristics and dictionaries.

At the same time, much work has been done on automated lexical acquisition for a single language, and in particular, on the web-based acquisition of various types of semantic relationships. There is a substantial amount of related studies which deal with the discovery of various relationship types represented in useful resources such as WordNet, including hypernymy (Pantel et al, 2004; Snow et al., 2006), synonymy (Davidov and Rappoport, 2006; Widdows and Dorow, 2002) and meronymy (Berland and Charniak, 1999; Girju et al, 2006). Since named entities are very important in NLP, many studies define and discover relations between named entities (Hassan et al., 2006). Work was also done on relations between verbs (Chklovski and Pantel, 2004). There is growing research on relations between nominals (Girju et al., 2007).

While the majority of studies focus on extracting pre-specified semantic relationships, several recent studies were done on the automated discovery of unspecified relationship types. Thus Turney (2006) provided a pattern distance measure that allows a fully unsupervised measurement of relational similarity between two pairs of words on the same language. Banko et al. (2007) and Rosenfeld and Feldman (2007) find relationship instances where the relationships are not specified in advance. (Davidov and Rappoport, 2008a) introduced the idea that salient semantic relationships can be defined as pattern clusters, confirming it with SAT analogy test. As explained above, we use this definition in the present study. We also use pattern clusters given by (Davidov and Rappoport, 2008a) as input in our evaluation.

Most of the relationship acquisition studies were done in a single language. Those that experiment in several languages usually treat each language separately, while we extract a relationship definition for one language using the provided definition for the other language.

Our study is related to cross-language information retrieval (CLIR) frameworks. Both deal with multilingual information extracted from the Web. However, the majority of CLIR studies pursue different targets. Thus, one of the main CLIR goals is the retrieval of *documents* based on explicit queries, when the document language is not the query language (Volk and Buitelaar, 2002). These frameworks usually develop language-specific tools and algorithms including parsers, taggers and morphology analyzers in order to integrate multilingual *queries* and *documents* (Jagarlamudi and Kumaran, 2007). Our goal is to develop and evaluate a *language-independent* algorithm for the *cross-lingual translation of relationship-defining structures*. While our targets are different from those of CLIR, CLIR systems can greatly benefit from our framework, since we can translate the relationships in CLIR queries and subsequently check if the same relationships are present in the retrieved documents.

Another field indirectly related to our research is Machine translation (MT). Many MT tasks require automated creation or improvement of dictionaries (Koehn and Knight, 2001). However, MT mainly deals with translation and disambiguation of words at the sentence or document level, while we translate relationship structures as a set of patterns, defined independently of contexts. We also perform pattern-set to pattern-set translation rather than the pattern-to-pattern or pair-to-pair translation commonly explored in MT studies. This makes it difficult to perform meaningful comparison to existing MT frameworks. However, the MT studies benefit from the proposed framework by enhancement and verification of translated relationship instances.

In (Davidov and Rappoport, 2009), we proposed a framework for automated cross-lingual concept mining. We incorporate several principles from this study including concept extension and disambiguation of query language (See Section 3.3). However our goals here are different since we target cross-lingual acquisition of relationship structures rather then concept term lists.

3 Relationship Translation Framework

Our framework has the following stages: (1) given a set of patterns in a source language defining some lexical relationship, we use the web to obtain source language term pairs participating in this relationship; (2) we automatically translate the obtained terms in each pair to the target language using available multilingual dictionaries; (3) we retrieve web snippets where these translations co-appear, disambiguating translations with web counts and extracting the corresponding patterns. As an optional final stage, the translated pattern cluster can be used to extract and extend a set of target language term pairs. Now we describe each of these stages in detail.

3.1 Acquisition of representative term pairs

We are provided with a pattern cluster, a set of patterns representing a specific lexical relationship in some language. The goal of the first stage is to discover the most representative term pairs for this cluster and language from the web. If the relationship is already specified by a representative set of term pairs, we skip this stage and continue to the next stage. Note that the method described below can also be used at the final stage to obtain representative *target* language term pairs once we obtain a *target* language pattern cluster.

The input lexical patterns are surface patterns which include several fixed words or punctuation symbols and two slots for content words, e.g. *"the [X] of the [Y],"*. Given a cluster of patterns defining a semantic relationship, we would like to obtain from the web the most representative and frequent examples of the represented relationship. In order to do that we construct search engine queries[2] from the given patterns using wildcard symbols to represent pattern slots. For example, given a pattern *"the [X] of the [Y],"* we construct queries such as *"the * of the"; "the * * of the"*[3]. We collect all the retrieved search engine snippets and extract the appropriate term pairs found in these snippets.

Now we would like to select the most useful of the extracted pairs. Since the obtained pairs are only useful if we can translate them into the target language, we dismiss all pairs in which one or both terms have no translations to the target language in our dictionaries (see Section 3.2). Since each particular pattern can be ambiguous, we also dismiss pairs which were found for only a single pattern in the given cluster.

For the remaining term pairs we would like to estimate their specificity for the given pattern cluster. For each pattern, we retrieve and use two web hit counts: $F_{terms}(p, T1, T2)$, a hit count for co-appearance of the pair in a way similar to that in the pattern, and $F_{all}(p, T1, T2)$, the hit count of the full pattern instance.

For example, if for the pattern *p="the * of the"* we obtain a term pair (CEO, company), then $F_{all}(p)$=*Hits("the CEO of the company")* and $F_{terms}(CEO, company)$= *Hits("CEO * * company")*. Given a pattern cluster C with patterns $\{p_1 \ldots p_n\} \in C$, we estimate the specificity of a term pair $(T1, T2)$ using the following simple probabilistic metric, giving to all patterns in the cluster an equal weight:

$$Spec(T1, T2) = \frac{1}{n} \sum_{p_i \in C} \frac{F_{all}(p_i, T1, T2)}{F_{terms}(p_i, T1, T2)}$$

We select the top 15 pairs with the highest specificity and use them in the next stage.

[2]We use Yahoo! Boss.

[3]Since the search engine API doesn't allow punctuation, we omit the punctuation in queries, but require a proper punctuation when processing the obtained snippet data.

3.2 Translation of the term pairs

After the previous stage we have a good representative set of term pairs for the desired source language relationship. Now we would like to translate the words in these pairs to the target language. In order to do that we use an extensive set of 1067 multilingual dictionaries developed for StarDict[4], including Wikipedia cross-language links and Wiktionary. For each term we obtain a set of its translations to the target language. If we get more than five different translations, we select the five having the highest number of dictionaries where this translation appears.

As discussed in Section 3.1, we dismissed terms for which no translation was found in any of the available dictionaries, so each term in each of the obtained pairs has at least a single translation to the target language. However, in many cases the available translations represent the wrong word sense, since both the source terms and their translations can be ambiguous. Thus at this stage many of the obtained term translations are irrelevant for the given relationship and require disambiguation.

3.3 Web mining for translation contexts

For this stage, we need to restrict web mining to specific target languages. This restriction is straightforward if the alphabet or term translations are language-specific or if the search API supports restriction to this language. In case where there is no such natural restrictions, we attempt to detect and add to our queries a few language-specific frequent words. Following (Davidov and Rappoport, 2009), we use our dictionaries to find 1–3 of the 15 most frequent words in a desired language[5] that are unique to that language and 'and' them with the queries to ensure proper language selection. This allows applying our algorithm to more than 60 diverse languages. The only data required for each language is at least a partial coverage of the obtained term pairs by some available dictionary.

Given a term pair $(T1, T2)$ we obtain a set of translations $(T1'_{i \in 1 \ldots n}, T2'_{j \in 1 \ldots m})$. For each combination $T1'_i, T2'_j$ of the obtained term translations, we construct and execute the following

[4]http://stardict.sourceforge.net/

[5]We estimated the word frequencies from text available in the corresponding multilingual dictionaries.

four queries: $\{$"$T1'_i \; * \; T2'_j$", "$T2'_j \; * \; T1'_i$", "$T1'_i \; * \; * \; T2'_j$", "$T2'_j \; * \; * \; T1'_i$"$\}$[6]. Since $Yahoo!Boss$ allows retrieval of up to the 1000 first results, we can collect up to four thousand snippets for each combination. However, the majority of these combinations return no snippets at all, effectively generating an average of a dozen snippets per query.

3.4 Pattern extraction

Now for each pair of term translations we would like to extract from the snippets all surface patterns which connect the terms in this pair. We use the basic two-slot meta-pattern type:

$$[Prefix] \; X \; [Infix] \; Y \; [Postfix]$$

X and Y should be the translated terms, Infix may contain punctuation, spaces, and up to four words (or up to eight symbols in languages without space-separated words like Chinese). Prefix and Postfix are limited to contain one or zero punctuation characters and/or up to two words. We do not allow empty Infix, Prefix of Postfix. If there are several possible combinations of Prefix and Postfix we generate a pattern set for all possible combinations (e.g., if we retrieve a snippet ...", *consider using [plexiglass] for [kitchen].*"..., we create patterns "using X for Y.", "consider using X for Y." and ", consider using X for Y.").

Now we would like to find the patterns representing the relationship in the target language. We do this in two stages. First we would like to detect the most common patterns for the given relationship. Let S_k be the union set of all patterns obtained for all combinations of the extracted translations for a specific source language term pair $k \in 1 \ldots K$. Let $Salience(p) = \frac{1}{K}|\{k|p \in S_k\}|$ be the portion of source language term pairs which lead to detection of the target language pattern p. We compute salience for each pattern, and select a subset of *salient patterns*, defined to be those whose Salience exceeds a predefined threshold (we used $1/3$). If one salient pattern is a substring of another salient pattern, we only select the longer one.

[6]These are Yahoo! queries where enclosing words in "" means searching for an exact phrase and "*" means a wildcard for exactly one arbitrary word.

In our salience estimation we mix data from all combinations of translations including incorrect senses and wrong translations of ambiguous terms. Now we would like to select a single correct target language pair for each source language pair in order to find more refined relationship representing patterns. For each source language term pair, we select the target language translated pair which captured the highest number of salient patterns. In case there are several pairs with the same number of salient patterns, we select a pair with the greatest web hit count. We drop term pairs with zero salient patterns.

Finally we would like to enhance the obtained set of salient patterns with more precise and representative relationship-specific patterns. Since we disambiguated the translated pairs, target language patterns captured by the remaining term pairs should be more trusted. We compare the target language pattern sets obtained for different remaining term pairs, and collect all patterns that were captured by at least three different term pairs. As before, if one pattern is a substring of another we retain only the longer one. As a result we get a comprehensive target language pattern cluster for the desired relationship.

3.5 Retrieval of target language term pairs

As an optional final stage, we can utilize the retrieved target language pattern clusters in order to discover target language term pairs for the desired relationship. We do this by utilizing the strategy described in Section 3.1 on the obtained target language pattern clusters. We do not dismiss obtained terms having no available dictionary translations, and we do not limit our search to the 15 terms with highest specificity. Instead we either select N term pairs with top specificity (where N is provided by user as in our evaluation), or we select all term pairs with specificity above some threshold.

4 Evaluation

In order to test the quality of the translated pattern clusters and the corresponding translated term pairs, we need to check both flexibility and correctness. Flexibility measures how well the retrieval works well *across languages* and for *many* *types* of semantic relationships. To do that, we tested our framework on both generic and specific relationships for 11 languages. Correctness verifies that the retrieved set of target language patterns and the corresponding term pairs represent *the same semantic relationship* as the given set of source language term pairs or patterns. To do that, we used both manual cross-lingual analogy-based correctness evaluation and evaluation based of factual data.

4.1 Languages and relationships

One of the main goals in this research was to provide a fully automated and flexible framework, which requires minimal modifications when applied to different languages and relationships.

We examined an extensive set of target languages using English as a source language. Table 1 shows 11 languages used in our experiments. We included west European languages, Slavic languages like Russian, Semitic languages like Hebrew, and Asian languages such as Chinese. We developed a set of tools for automatic off-line access to an extensive set of 1067 multilingual dictionaries created for the StarDict platform. These dictionaries include recent dumps of Wikipedia cross-language links and Wiktionary data.

In our experiments we used three sets of relationships: **(1) Generic:** 15 unsupervisedly discovered English pattern clusters representing various generic relationships. **(2) H-M-C:** The three most studied relationships: hypernymy, meronymy and co-hyponymy.**(3) Specific:** Three factual relationships: country-capital, country-language and dog breed-origin. Below we describe the evaluation of each of these sets in detail. Note that our framework allows two ways of specifying a source language relationship – a pattern cluster and a set of term pairs.

4.2 Evaluation of generic pattern clusters

In our **Generic** evaluation setting, we utilized as input a random sample of 15 automatically discovered relationship definitions. We started from a set of 508 English pattern clusters, unsupervisedly discovered using the method of (Davidov and Rappoport, 2008a). Each of these clusters is assumed to represent a distinct semantic relationship. We randomly selected 15 pattern clusters from this set and executed our framework on these clusters to obtain the corresponding target language pattern clusters for each of the 11 tested languages. An example of a partial set of patterns in a cluster is: *'this [X] was kept in [Y],';'the X that he kept in [Y],';'the [X] in the [Y] and';'the [Y] containing the [X]'*. . .

We then used the term pair selection algorithm described in Section 3.1 to select the most specific term pair for each of the 15 source language clusters and 10 pairs for each of the corresponding translated target language clusters. Thus for each of the 15 pattern clusters and for each of the 11 languages we produced a single source language term pair and up to 10 corresponding target language term pairs.

In order to check the correctness of translation of an unspecified semantic relationship we need to compare source and target language relationships. Comparison of relationships is a challenging task, since there are no relationship resources for most relationship types even in a single language, and certainly so for their translations across languages. Thus various studies define and split generic relationships differently even when describing relatively restricted relationship domains (e.g., relationships holding between parts of noun phrases (Nastase and Szpakowicz, 2003; Moldovan et al., 2004)). In order to compare generic relationships we used a manual cross-lingual SAT-like analogy human judgment evaluation[7]. This allowed us to assess the quality of the translated pattern clusters, in a similar way as (Davidov and Rappoport, 2008a) did for testing clusters in a single language.

For each of the 15 clusters we constructed a cross-lingual analogy question in the following manner. The header of the question was a term pair obtained for the source language pattern cluster. The six multiple choice items included: (1) one of the 10 discovered translated term pairs of the same cluster (the 'correct' answer)[8]; (2) three

[7] Using Amazon's Mechanical Turk.

[8] We avoid selection of the target language pairs which were obtained through direct translation of the source language pair given at the header of the question. This is crucial so that subjects will not judge correctness of translation but correctness of the relationship.

of the translated pairs of the other clusters among the 15; (3) a pair constructed by randomly selecting terms from different translated clusters; (4) the 6th option states that either the given options include broken words or incorrect language, or none of the presented pairs even remotely exemplifies the relationship in question. An example question for English-Italian:

The English pair: *(kennel, dog)*; (1) "correct" pair: *(acquario, pesce)*; (2)-(4) "wrong" pairs: *(topo, orecchio), (mela, rossa), (occhio, grande)*; (5) "random": *(scodella, scatola)*; (6) Pairs comprise non-Italian/broken words or no pair exemplifi es the relationship

In order to check the English proficiency of the subjects we added 5 "easy" monolingual English SAT analogy questions. We also added a single hand-crafted cross-lingual question of an obvious analogy case, making a total of 16 cross-lingual questions. Subjects who failed more than one of the easy English SAT questions or failed the obvious cross-lingual question were rejected from the evaluation. Finally we have three subjects for each of the tested languages. We also asked the subjects to assign a confidence score from 0 (worst) to 10 (best) to express how well the selected term pair represents the source language relationship in question.

Language	P	% 6th	$Score_c$	$Score_w$
Chinese	71	9	9.1	1.8
Czech	73	9	8.3	2.0
French	80	10	8.4	1.9
German	68	9	8.3	1.5
Greek	72	11	8.7	2.0
Hebrew	69	11	9.0	2.5
Hindi	62	12	7.4	1.9
Italian	70	10	8.5	1.5
Russian	75	8	9.0	1.6
Turkish	61	13	9.1	2.0
Ukrainian	73	11	9.3	2.3
Average	70	10	9.1	1.9

Table 1: Averaged results for manual evaluation of 15 pattern clusters. P: precision (% of correct answers); % 6th: percentage of 6th selection; $Score_c$: averaged confi dence score for correct selections; $Score_w$: confi dence score for wrong selections.

We computed accuracy and agreement for the given answers (Table 1). We can see that for all languages above 61% of the choices were correct (comparing to 75% reported by (Davidov and Rappoport, 2008a) for a similar *monolingual* analogy test for the same set of pattern clusters). While the results are obviously lower than the corresponding single-language test, they are significantly above the random baseline of 20%[9]. Also note that as reported in (Turney, 2006), an average single-language highschool SAT grade is 57, which is lower than the scores obtained for our cross-lingual test. We can also see that for the correctly selected pairs the confidence score was very high, while the score for wrongly selected pairs was significantly lower.

4.3 Evaluation of the H-M-C relationships

In order to test how well our algorithm performs on the most common and useful relationships, hypernymy, meronymy and co-hyponymy, we automatically sampled from WordNet a set of 10 source language term pairs for each of these relationships and applied our framework to extract up to 100 target language term pairs for each of the three relationships as done above.

For each of the tested languages we presented to three human subjects for each language a short English definition of hypernymy, meronymy and co-hyponymy, along with the corresponding randomly selected 10 of 100 extracted pairs, and asked them to rank how well (0 (worst) to 10 (best)) each pair represents the described relationship. In order to reduce possible bias, we mixed in each set 3 randomly selected term pairs obtained for the other two relationships. Table 2 shows the average scores for this task.

Language	Hypernymy	Meronymy	Co-hyponymy	Random
Chinese	8.0	7.1	8.1	1.9
Czech	8.4	7.0	8.5	2.3
French	8.1	7.5	8.4	1.8
German	8.4	7.1	8.6	2.4
Greek	8.7	7.5	8.6	1.8
Hebrew	8.6	7.9	8.3	1.6
Hindi	7.5	7.1	7.8	2.2
Italian	7.9	7.8	8.2	1.5
Russian	8.6	8.1	8.9	1.7
Turkish	8.3	7.2	8.6	1.7
Ukrainian	8.2	7.7	8.2	1.7
Average	8.3	7.5	8.4	1.9

Table 2: Averaged results for hypernymy, meronymy and co-hyponymy translations. The three fi rst columns show average scores for hypernymy, meronymy and co-hyponymy relationships. The last column shows scores for the random baseline.

We can see that our algorithm successfully detects the common relationships, achieving high scores. Also the results indicate that the patterns

[9] A reasonable random baseline omits the 6th option.

are sufficiently precise to extract at least 100 of the instances for the given salient relationships.

4.4 Evaluation of the specific relationships

To check how well our algorithm performs on some specific relationships, we examined its performance on three specific relationships explored in previous studies. We provided it with 10 source language (English) term pair examples for each of the (country, capital), (country, language) and (dog breed, origin) relationships. For each of these relationships we have factual information for every tested target language available through Wikipedia list articles. This allows us to perform an unbiased automated evaluation of the quality of the obtained target language data.

We applied our framework on these examples and generated 30 target language pairs with highest specificity for each of these relationships and languages. We compared the retrieved pairs to the factual data. Table 3 shows the precision of the results obtained for these patterns.

Language	Capital	Language	Dog breed
Chinese	0.87	0.83	0.8
Czech	0.93	0.83	0.77
French	0.97	0.9	0.87
German	0.93	0.9	0.83
Greek	0.87	0.83	0.77
Hebrew	0.83	0.8	0.8
Hindi	0.83	0.8	0.77
Italian	0.93	0.87	0.83
Russian	0.97	0.9	0.87
Turkish	0.87	0.83	0.83
Ukrainian	0.93	0.87	0.8
Average	0.9	0.85	0.81

Table 3: Precision for three specific relationship types: (country, capital), (country, language) and (dog breed,origin).

The precision observed for this task is comparable to precision obtained for Country-Capital and Country-Language in a previous single-language acquisition study (Davidov et al., 2007)[10]. The high precision observed for this task indicates that the obtained translated patterns are sufficiently good as a seed for pattern-based mining of specific relationships.

[10]It should be noted however that unlike previous work, we only examine the first 30 pairs and we do not use additional disambiguating words as input.

5 Conclusion

We proposed a framework which given a set of patterns defining a semantic relationship in a specific source language uses multilingual dictionaries and the web to discover a corresponding pattern cluster for a target language. In the evaluation we confirmed the applicability of our method for different languages and relationships.

The obtained set of target language pattern clusters can be used for acquisition of relationship instances as shown in our evaluation. An interesting direction for future work is to use the discovered target language pattern clusters in NLP tasks like textual entailment which require distinguishing between semantic relationships.

Applying our framework to the set of unsupervisedly discovered relationships allows a fully automated construction of a relationship dictionary, where pattern clusters in one language correspond to patten clusters in many other languages. Unlike the majority of existing machine translation systems, construction of this dictionary does not require parallel corpora. Such a dictionary can be useful for machine translation, cross-lingual textual entailment and query translation, to name just a few applications. In the future we plan to create a multilingual pattern cluster dictionary which interconnects pattern clusters from many languages and allows cross-lingual definition of lexical relationships.

References

Amasyali Fatih, 2005. Automatic Construction of Turkish Wordnet. *Signal Processing and Communications Applications Conference.*

Mishele Banko, Michael Cafarella , Stephen Soderland, Matt Broadhead, Oren Etzioni, 2007. Open information extraction from the Web. *IJCAI '07.*

Matthew Berland, Eugene Charniak, 1999. Finding parts in very large corpora. *ACL '99.*

Thatsanee Charoenporn, Virach Sornlertlamvanich, Chumpol Mokarat, and Hitoshi Isahara, 2008. Semi-automatic Compilation of Asian WordNet. *Proceedings of the 14th NLP-2008, University of Tokyo, Komaba Campus, Japan.*

Timothy Chklovski, Patrick Pantel, 2004. VerbOcean:

mining the web for fine-grained semantic verb relations. *EMNLP '04*.

Dmitry Davidov, Ari Rappoport, 2006. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. *COLING-ACL '06.*

Dmitry Davidov, Ari Rappoport and Moshe Koppel, 2007. Fully Unsupervised Discovery of Concept-Specific Relationships by Web Mining. *ACL '07.*

Dmitry Davidov, Ari Rappoport. 2008a. Unsupervised Discovery of Generic Relationships Using Pattern Clusters and its Evaluation by Automatically Generated SAT Analogy Questions. *ACL '08.*

Dmitry Davidov and Ari Rappoport, 2008b. Classification of relationships between nominals using pattern clusters. *ACL '08*.

Dmitry Davidov and Ari Rappoport, 2009. Translation and Extension of Concepts Across Languages. *EACL '09*.

Roxana Girju, Adriana Badulescu, and Dan Moldovan, 2006. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1).

Roxana Girju, Marthy Hearst, Preslav Nakov, Vivi Nastase, Stan Szpakowicz, Peter Turney and Yuret, D., 2007. Task 04: Classification of semantic relations between nominal at SemEval 2007. *4th Intl. Workshop on Semantic Evaluations (SemEval '07), in ACL '07*.

Hany Hassan, Ahmed Hassan and Ossama Emam, 2006. Unsupervised information extraction approach using graph mutual reinforcement. *EMNLP '06*.

Jagadeesh Jagarlamudi, A Kumaran, 2007 Cross-Lingual Information Retrieval System for Indian Languages *Working Notes for the CLEF 2007 Workshop.*

Philipp Koehn and Kevin Knight. 2001. Knowledge sources for word-level translation models. *EMNLP '01*.

Dan Moldovan, Adriana Badulescu, Marta Tatu, Daniel Antohe, and Roxana Girju, 2004. Models for the semantic classification of noun phrases. *HLT-NAACL '04 Workshop on Computational Lexical Semantics.*

Vivi Nastase, Stan Szpakowicz, 2003. Exploring noun-modifier semantic relations. *In Fifth Intl. Workshop on Computational Semantics (IWCS-5)*.

Patrick Pantel, Deepak Ravichandran, Eduard Hovy, 2004. Towards terascale knowledge acquisition. *COLING '04*.

Marius Pasca, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, Alpa Jain, 2006. Names and similarities on the web: fact extraction in the fast lane. *COLING-ACL 06*.

Adam Pease, Christiane Fellbaum, Piek Vossen, 2008. Building the Global WordNet Grid. *CIL18*.

Benjamin Rosenfeld , Ronen Feldman, 2007. Clustering for unsupervised relation identification. *CIKM '07*.

Rion Snow, Daniel Jurafsky, Andrew Ng, 2006. Semantic taxonomy induction from heterogeneous evidence. *COLING-ACL '06*.

Peter Turney, 2005. Measuring semantic similarity by latent relational analysis, *IJCAI '05*.

Peter Turney, 2006. Expressing implicit semantic relations without supervision. *COLING-ACL '06*.

Martin Volk, Paul Buitelaar, 2002 A Systematic Evaluation of Concept-Based Cross-Language Information Retrieval in the Medical Domain. *In: Proc. of 3rd Dutch-Belgian Information Retrieval Workshop.* Leuven.

Dominic Widdows, Beate Dorow, 2002. A graph model for unsupervised Lexical acquisition. *COLING '02*.

Comparison of different algebras for inducing the temporal structure of texts

Pascal Denis[†]
† Alpage Project-Team
INRIA & Université Paris 7
pascal.denis@inria.fr

Philippe Muller[†,◇]
◇ IRIT
Université de Toulouse
muller@irit.fr

Abstract

This paper investigates the impact of using different temporal algebras for learning temporal relations between events. Specifically, we compare three interval-based algebras: Allen (1983) algebra, Bruce (1972) algebra, and the algebra derived from the TempEval-07 campaign. These algebras encode different granularities of relations and have different inferential properties. They in turn behave differently when used to enforce global consistency constraints on the building of a temporal representation. Through various experiments on the TimeBank/AQUAINT corpus, we show that although the TempEval relation set leads to the best classification accuracy performance, it is too vague to be used for enforcing consistency. By contrast, the other two relation sets are similarly harder to learn, but more useful when global consistency is important. Overall, the Bruce algebra is shown to give the best compromise between learnability and expressive power.

1 Introduction

Being able to recover the temporal relations (e.g., precedence, inclusion) that hold between events and other time-denoting expressions in a document is an essential part of natural language understanding. Success in this task has important implications for other NLP applications, such as text summarization, information extraction, and question answering.

Interest for this problem within the NLP community is not new (Passonneau, 1988; Webber, 1988; Lascarides and Asher, 1993), but has been recently revived by the creation of the TimeBank corpus (Pustejovsky et al., 2003), and the organization of the TempEval-07 campaign (Verhagen et al., 2007). These have seen the development of machine learning inspired systems (Bramsen et al., 2006; Mani et al., 2006; Tatu and Srikanth, 2008; Chambers and Jurafsky, 2008).

Learning the temporal stucture from texts is a difficult problem because there are numerous information sources at play (in particular, semantic and pragmatic ones) (Lascarides and Asher, 1993). An additional difficulty comes from the fact that temporal relations have logical properties that restrict the consistent graphs that can be built for a set of temporal entities (for instance the transitivity of inclusion and temporal precedence). Previous work do not attempt to directly predict globally coherent temporal graphs, but instead focus on the the simpler problem of labeling pre-selected pairs of events (i.e., a task that directly lends itself to the use of standard classification techniques). That is, they do not consider the problem of *linking* pairs of events (i.e., of determining which pairs of events are related).

Given the importance of temporal reasoning for determining the temporal structure of texts, a natural question is how to best use it within a machine-based learning approach. Following (Mani et al., 2006), prior approaches exploit temporal inferences to enrich the set of training instances used for learning. By contrast, (Bramsen et al., 2006) use temporal relation compositions to provide constraints in a global inference problem (on the slightly different task of ordering passages in medical history records). (Tatu and Srikanth, 2008) and (Chambers and Jurafsky, 2008) combine both approaches and use temporal reasoning both during training and decoding. Interestingly, these approaches use different inventories of relations: (Mani et al., 2006) use the TimeML 13 relation set, while (Chambers and Jurafsky, 2008;

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 250–258,
Beijing, August 2010

Bramsen et al., 2006) use subset of these relations, namely precedence and the absence of relation.

This paper adopts a more systematic perspective and directly assesses the impact of different relation sets (and their underlying algebras) in terms of learning and inferential properties. Specifically, we compare three interval-based algebras for building classification-based systems, namely: Allen (1983)'s 13 relation algebra, Bruce (1972)'s 7 relations algebra, and the algebra underlying Tempeval-07 3 relations (henceforth, TempEval algebra). We wish to determine the best trade-off between: (i) how easy it is to learn a given set of relations, (ii) how informative are the representations produced by each relation set, and (iii) how much information can be drawn from the predicted relations using knowledge encoded in the representation. These algebras indeed differ in the number of relations they encode, and in turn in how expressive each of these relations is. From a machine learning point of view of learning, it is arguably easier to learn a model that has to decide among fewer relations (i.e., that has fewer classes). But from a representational point of view, it is better to predict relations that are as specific as possible, for composing them may restrict the prediction to more accurate descriptions of the situation. However, while specific relations potentially trigger more inferences, they are also more likely to predict inconsistent constraints. In order to evaluate these differences, we design a set of experiments on the Timebank/AQUAINT corpus, wherein we learn precise relations and vaguer ones, and evaluate them with respect to each other (when a correspondence is possible).

Section 2 briefly presents the Timebank/AQUAINT corpus. In section 3, we describe the task of temporal ordering through an example, and discuss how it should be evaluated. Section 4 then goes into more detail about the different representation possibilities for temporal relations, and some of their formal properties. Section 5 presents our methods for building temporal structures, that combines relation classifiers with global constraints on whole documents. Finally, we discuss our experimental results in section 6.

2 The Timebank/AQUAINT corpus

Like (Mani et al., 2006) and (Chambers and Jurafsky, 2008), we use the so-called OTC corpus, a corpus of 259 documents obtained by combining the Timebank corpus (Pustejovsky et al., 2003) (we use version 1.1 of the corpus) and the AQUAINT corpus.[1] The Timebank corpus consists of 186 newswire articles (and around $65,000$ words), while AQUAINT has 73 documents (and around $40,000$ words).

Both corpora are annotated using the TimeML scheme for tagging eventualities (events and states), dates/times, and their temporal relations. Eventualities can be denoted by verbs, nouns, and some specific constructions. The temporal relations (i.e., the so-called TLINKS) encode topological information between the time intervals of occurring eventualities. TimeML distinguishes three types of TLINKS: event-event, event-time, and time-time, giving rise to different subtasks. In this paper, we will focus on predicting event-event relations (see (Filatova and Hovy, 2001; Boguraev and Ando, 2005) for work on the other tasks). The set of temporal relations used in TLINKS mirrors the 13 Allen relations (see next section), and includes the following six relations: *before*, *begins*, *ends*, *ibefore*, *includes*, *simultaneous* and their inverses. The combined OTC corpus comprises a total of $6,139$ annotated event-event TLINKS. We also make use of the additional TLINKS independently provided by (Bethard et al., 2007) for 129 of the 186 Timebank documents.

3 Task presentation and evaluation

3.1 An example

We illustrate the task of event ordering using a small fabricated, simplified example:

> Fortis bank <u>invested</u>$_{e_1}$ in junk bonds before <u>the financial crisis</u>$_{e_2}$, but <u>got rid</u>$_{e_3}$ of most of them during <u>the crisis</u>$_{e_{2bis}}$. However, the institution still <u>went bankrupt</u>$_{e_4}$ a year later.

[1] Both corpora are freely available from `http://www.timeml.org/site/timebank/timebank.html`.

The annotation for this temporal structure would include the following relations: e_1 is temporally before e_2, e_3 is temporally included in e_2, and e_3 is before e_4. The coreference relation between e_2 and e_{2bis} implies the equality of their temporal extension. Of course all these events may in theory be related temporally to almost any other event in the text. Events are also anchored to temporal expressions explicitly, and this is usually considered as a separate, much easier task. We will use this example throughout the rest of our presentation.

3.2 Comparing temporal annotations

Due to possible inferences, there are often many equivalent ways to express the same ordering of events, so comparisons between annotation and reference event-event pairs cannot rely on simple precision/recall measures.

Consider the above example and assume the following annotation: e_1 is before e_2, e_3 is included in e_2, and e_3 is before e_3. Without going into too much detail about the semantics of the relations used, one expects annotators to agree with the fact that it entails that e_1 is before e_3, among other things. So the annotation is equivalent to a larger set of relations. In some cases, the inferred information is disjunctive (the relation holding between two events is a subset of possible "simple" relations, such as "before or included").

Nowadays, the given practice is to compute some sort of transitive closure over the network of constraints on temporal events (usually expressed in the well-studied Allen algebra (Allen, 1983)), and compute agreements over the saturated structures. Specifically, we can compare the sets of *simple* temporal relations that are deduced from it (henceforth, the "strict" metric), or measure the agreement between the whole graphs, including disjunctions (Verhagen et al., 2007) (henceforth, the "relaxed" metric).[2] Under this latter metric, precision (resp. recall) of a prediction for a pair of events consisting of a set S of relations with respect to a set of relations R inferred from the reference, is computed as $|S \cap R|/|S|$ (resp. $|S \cap R|/|R|$).

[2]Taking into account disjunctions means giving partial credit to disjunctions approximating the reference relation (possibly disjunctive itself), see next section.

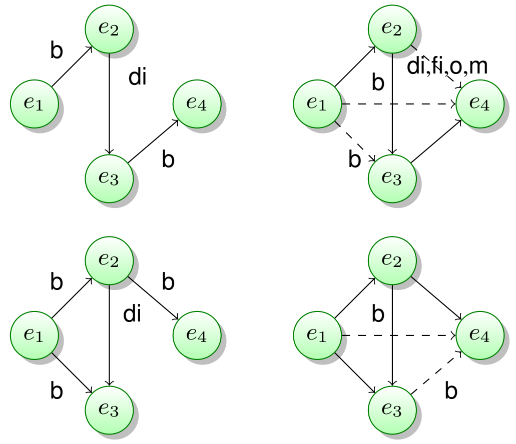


Figure 1: Two non-equivalent annotations of the same situations (left) and their transitive closure in Allen's algebra (right, with new relations only). b stands for Allen's *before* relation, m for *meet*, o for *overlap*, di and fi for the inverses of *during* and *finish*, respectively.

Figure 1 illustrates the point of these "saturated" representations, showing two raw annotations of our example on the left (top and bottom) and their closures on the right. The raw annotations share only 2 relations (between e_1 and e_2, and e_3 and e_4), but their transitive closures agree also on the relations between e_1 and e_3, e_1 and e_4, and e_3 and e_4. They still differ on the relation between e_2 and e_4, but only because one is much more specific than the other, something that can only be taken into account by a partial credit scoring function.

For this example, the "strict" metric yields precision and recall scores of 5/5 and 5/6, when comparing the top annotation against the bottom one. By contrast, the "relaxed" metric (introduced in the TempEval-07) yields precision and recall scores of (5+0.2)/6 and 6/6, respectively.

We now turn to the issue of the set of relations chosen for the task of expressing temporal information in texts.

4 Temporal representations

Because of the inferential properties of temporal relations, we have seen that the same situation can be expressed in different ways, and some relations can be deduced from others. The need for a precise reasoning framework has been present in previous attempts at the task (Setzer et al., 2006), and people have moved to a set of hand-made rules over ad hoc relations to more widely accepted temporal reasoning frameworks, such as algebras of temporal relations, the most famous being Allen's interval algebra.

An algebra of relations can be defined on any set of relations that are mutually exclusive (two relations cannot hold at the same time between two entities) and exhaustive (at least one relation must hold between two given entities). The algebra starts from a set of simple, atomic, relations $U = \{r_1, r_2, ...\}$, and a general relation is a subset of U, interpreted as a disjunction of the relations it contains. From there, we can define union and intersection of relations as classical set union and intersection of the base relations they consist of. Moreover, one can define a composition of relations as follows:

$$(r_1 \circ r_2)(x, z) \leftrightarrow \exists y \; r_1(x, y) \wedge r_2(y, z)$$

In words, a relation between x and z can be computed from what is known between (x and y) and (y and z). By computing beforehand the $n \times n$ compositions of base relations of U, we can compute the composition of any two general relations (because $r \cap r' = \emptyset$ when r, r' are basic and $r \neq r'$):

$$\{r_1, r_2, ...r_k\} \circ \{s_1, s_2, ...s_m\} = \bigcup_{i,j} (r_i \circ s_j)$$

Saturating the graph of temporal constraints means applying these rules to all compatible pairs of constraints in the graph and iterating until a fixpoint is reached. In Allen's algebra there are 13 relations, determined by the different relations that can hold between two intervals endpoints (before, equals, after). These relations are: b (*before*), m (*meet*), o (*overlap*), s (*start*), f (*finish*), d (*during*), their inverses (bi, mi, oi, si, fi, di) and = (*equal*), see figure 2.[3]

It is important to see that a general approach to temporal ordering of events cannot restrict itself to a subset of these and still use the power of inferences to complete a situation, because composition of information is stable only on restricted subsets. And using all of them means generating numerous disjunctions of relations.

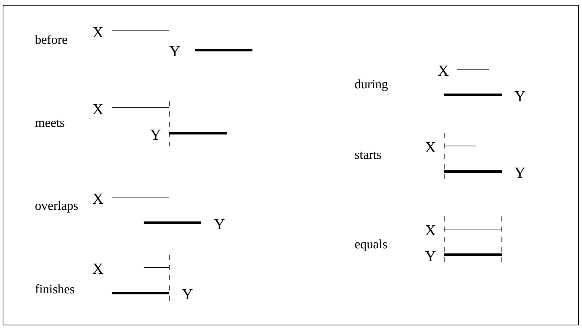


Figure 2: Allen's thirteen relations between two temporal intervals

Allen relations are convenient for reasoning purposes, but might too precise for representing natural language expressions, and that's why recent evaluation campaigns such as TempEval-07 have settled on vaguer representations. TempEval-07 uses three relations called *before*, *overlaps* and *after*, which we note b_t, o_t, and bi_t.[4] These all correspond to disjunctions of Allen relations: $\{\mathsf{b},\mathsf{m}\}_a$, $\{\mathsf{o},\mathsf{d},\mathsf{s},=,\mathsf{f}\}_a$ and its inverse, and $\{\mathsf{bi},\mathsf{mi}\}_a$, respectively. These representations can be converted to Allen relations, over which the same inference procedures can be applied, and then expressed back as (potentially disjunctive) TempEval relations. They thus form a sub-algebra of Allen's algebra, if we add their possible disjunctions.

In fact, starting from the base relations, only $\{\mathsf{b},\mathsf{o}\}_t$, $\{\mathsf{bi},\mathsf{o}\}_t$, and *vague* (i.e., the disjunction of all relations) can be inferred (besides the base relations). This is a consequence of the stability of so-called convex relations in Allen algebra. Note that an even simpler schema is used in (Chambers and Jurafsky, 2008), where only TempEval *before* and *after* and the *vague* relation are used.

We propose to consider yet another set of relation, namely relations from (Bruce, 1972). These provide an intermediate level of representation, since they include 7 simple relations. These are

[3] TimeML uses somewhat different names, with obvious mappings, except *ibefore* ("immediately before") for m, and *iafter* ("immediately after") for mi.

[4] When it is not obvious, we will use subscript symbols to indicate the particular algebra that is used (e.g., b_t is the before relation in TempEval).

also expressible as disjunctions of Allen relations; they are: *before* (b_b), *after* (bi_b) (with the same semantics as TempEval's b_t and bi_t), *equals* ($=_b$, same as $=_a$), *includes* (i, same as Allen's $\{\mathsf{s,d,f}\}_a$), *overlaps* (o_b, same as o_a), *included* (ii) and *is-overlapped* (oi_b), their inverse relations. The equivalences between the three algebras is shown table 1.

<table>
<tr><td>Allen</td><td>Bruce</td><td>Tempeval</td></tr>
<tr><td>before
meet</td><td>before</td><td>before</td></tr>
<tr><td>overlaps</td><td>overlaps</td><td rowspan="4">overlaps</td></tr>
<tr><td>starts
during
finishes</td><td>included</td></tr>
<tr><td>overlapsi</td><td>is-overlapped</td></tr>
<tr><td>startsi
duringi
finishesi</td><td>includes</td></tr>
<tr><td>meeti
beforei</td><td>after</td><td>after</td></tr>
<tr><td>equals</td><td>equals</td><td>equals</td></tr>
</table>

Table 1: Correspondances between temporal algebras. A relation ranging over multiple cells is equivalent to a disjunction of all the relations within these cells.

Considering a vaguer set is arguably more adequate for natural language expressions while at the same time this specific set preserves at least the notions of temporal order and inclusion (contrary to the TempEval scheme), which have strong inferential properties: they are both transitive, and their composition yields simple relations; overlap allows for much weaker inferences. Figure 3 shows part of our example from the introduction expressed in the three cases: with Allen relations, the most precise, with Bruce relations and TempEval relations, with dotted lines showing the extent of the vagueness of the temporal situations in each case (with respect to the most precise Allen description). We can see that TempEval relations lose quickly all information that is not before or after, while Bruce preserves inference combining precedence and temporal inclusion.

Information can be converted from one algebra to the other, since vaguer algebras are based on relations equivalent to disjunctions in Allen algebra. But conversion from a precise relation to a vaguer one and back to a more precise algebra leads to information loss. Hence on figure 3, the original Allen relation: $e_3\ \mathsf{d}_a\ e_2$ is converted to: $e_3\ \mathsf{o}_t\ e_2$ in TempEval, which converts back into the much less informative: $e3\ \{\mathsf{o}, \mathsf{d}, \mathsf{s}, =, \mathsf{f}, \mathsf{oi}, \mathsf{si}, \mathsf{fi}, \mathsf{di}\}_a\ e_2$. We will use these translations during our system evaluation to have a common comparison point between representations.

5 Models

5.1 Algebra-based classifiers

In order to compare the impact of the different algebras described in section 4, we build three event pair classification models corresponding to each relation set. The resulting Allen-based, Bruce-based, and Tempeval-based models therefore contain 13, 7, and 3 class labels, respectively.[5] For obvious sparsity issues, we did not include classes corresponding to disjunctive relations, as there are $2^{|R|}$ possible disjunctions for each relation set R.

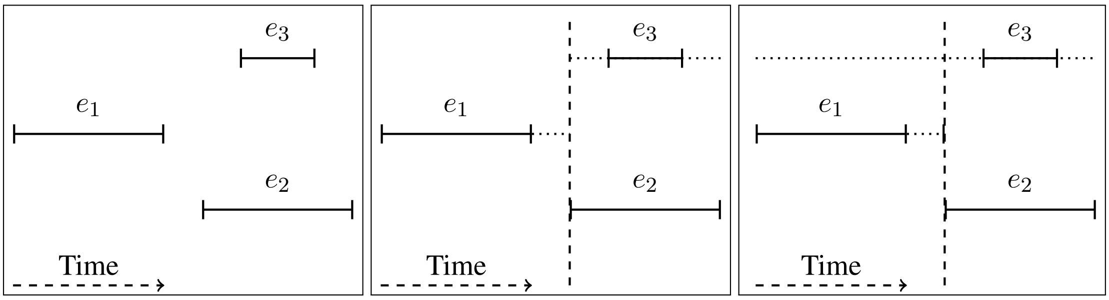
For training our models, we experiment with 4 various configurations that correspond to ways of expanding the set of training examples. Specifically, these configurations vary in: (i) whether or not we added the additional "Bethard relations" to the initial OTC annotations (Bethard et al., 2007), (ii) whether or not we applied saturation over the set of annotated relations.

5.2 Features

Our feature set for the various models is similar to that used by previous work, including binary features that encode event string as well as the five TimeML attributes and their possible values:

- **aspect**: none, prog, perfect, prog perfect
- **class**: report, aspectual, state, I-state I-action, perception, occurrence
- **modality**: none, to, should, would, could can, might
- **polarity**: positive, negative
- **tense**: none, present, past, future

[5] Our TempEval model actually has a fourth label for the *identity* relation. The motivations behind the inclusion of this extra label are: (i) this relation is linguistically motivated and comparatively easy to learn (for a lot of instances of this relation are cases of anaphora, which are often signaled by identical strings) (ii) this relation triggers a lot of specific inferences.



(a) Allen: $(e_1 \mathsf{b}_a e_2 \wedge e_3 \mathsf{d}_a e_2) \rightarrow e_1 \mathsf{b}_a e_3$

(b) Bruce: $(e_1 \mathsf{b}_b e_2 \wedge e_3 \mathsf{d}_b e_2) \rightarrow e_1 \mathsf{b}_b e_3$

(c) Tempeval: $(e_1 \mathsf{b}_t e_2 \wedge e_3 \mathsf{o}_t e_2) \rightarrow e_1 \{\mathsf{b}_t, \mathsf{o}_t\} e_3$

Figure 3: Comparing loss of inferential power in algebras: hard lines show the actual temporal model, exactly expressed in Allen relations (a); dotted lines show the vagueness induced by alternative schemes, and the inference that can or cannot still be made in each algebra, (b) and (c).

Additional binary features check agreement for same attribute (e.g., the same tense). Finally, we add features that represent the distance between two events (in number of sentences, and in number of intervening events). [6]

5.3 Training set generation

Our generic training procedure works as follows. For each document, we scan events in their order of appearance in the text. We create a training instance $inst_{(e_i,e_j)}$ for each *ordered* pair of events (e_i, e_j): if (e_i, e_j) (resp. (e_j, e_i)) corresponds to an annotated relation r, then we label $inst_{(e_i,e_j)}$ with the label r (resp. its inverse r^{-1}).

5.4 Parameter estimation

All of these classifiers are maximum entropy models (Berger et al., 1996). Parameter estimation was performed with the Limited Memory Variable Metric algorithm (Malouf, 2002) implemented in the Megam package.[7]

5.5 Decoding

We consider two different decoding procedures. The first one simply mirrors the training procedure just described, scanning pairs of events in the order of the text, and sending each pair to the classifier. The pair is then labeled with the label outputted by the classifier (i.e., the label receiving the highest probability). No attempt is made to guarantee the consistency of the final temporal graph.

Our second inference procedure works as follows. As in the previous method, we scan the events in the order of the text, and create ordered pairs of events that we then submit to the classifier. But the difference is that we saturate the graph after each classification decision to make sure that the graph created so far is coherent. In case where the classifier predicts a relation whose addition results in an incoherent graph, we try the next highest probability relation, and so on, until we find a coherent graph. This greedy procedure is similar to the Natural Reading Order (NRO) inference procedure described by (Bramsen et al., 2006).

6 Experiments and results

We perform two main series of experiments for comparing our different models. In the first series, we measure the accuracy of the Allen-, Bruce-, and Tempeval-based models on predicting the correct relation for the event-event TLINKS annotated in the corpus. In the second series, we saturate the event pair relations produced by the classifiers (combined with NRO search to enforce global coherence) and compare the predicted graphs against the saturated event-event TLINKS.

6.1 Experiment settings

All our models are trained and tested with 5-fold cross-validation on the OTC documents. For eval-

[6]These were also encoded as binary features, and the various feature values were binned in order to avoid sparseness.

[7]Available from `http://www.cs.utah.edu/~hal/megam/`.

uation, we use simple accuracy for the first series of experiments, and two "strict" and "relaxed" precision/recall measures described in section 3 for the other series. For each type of measures, we report scores with respect to both Allen and TemEval relation sets. All scores are reported using macro-averaging. Out of the 259 temporal graphs present in OTC, we found that 54 of them were actually inconsistent when saturated; the corresponding documents were therefore left out of the evaluation.[8] Given the rather expensive procedure involved in the NRO decoding (saturating an inconsistent graph "erases" all relations), we skipped 8 documents wich were much longer than the rest, leaving us with 197 documents for our final experiments.

6.2 Event-event classification

Table 2 summarizes the accuracy scores of the different classifiers on the event-event TLINKS of OTC. We only report the best configuration for each model. For the TempEval-based model, we found that the best training setting was when Bethard annotations were added to the original TimeML annotations, but with no saturation.[9] For Allen and Bruce models, neither Bethard's relations nor saturation helps improve classification accuracy. In fact, saturation degrades performance, which can be explained by the fact that saturation reinforces the bias towards already over-represented relations.[10] The best accuracy performances are obtained by the Allen-based and TempEval-based classifiers, each one performing better in its own algebra (with 47.0% and 54.0%). This is not surprising, since these classifiers were specifically trained to optimize their respective metrics. The Bruce-based classifier is slightly better than the Allen-based one in TempEval, but also slightly worse than TempEval-based classifier in Allen.

[8]Because there is no way to trace the relation(s) responsible for an inconsistency without analysing the whole set of annotations of a text, and considering that it usually happens on very long texts, we did not attempt to manually correct the annotations.

[9]This is actually consistent with similar findings made by (Chambers and Jurafsky, 2008).

[10]For instance, for Allen relations, there are roughly 50% of *before-after* relations before saturation but 73% of them after saturation.

	Allen Acc.	TempEval Acc.
Allen	47.0	48.9
Bruce	N/A	49.3
TempEval	N/A	54.0

Table 2: Accuracy scores for Allen, Bruce, and TempEval classifiers on event-event TLINKS, expressed in Allen or TempEval algebra. Scores for Bruce and TempEval models into Allen are left out, since they predict (through conversion) disjunctive relations for all relations but equality.

Our accuracy scores for Allen, and TempEval-based classifiers are somewhat lower than the ones reported for similar systems by (Mani et al., 2006) and (Chambers and Jurafsky, 2008), respectively. These differences are likely to come from the fact that: (i) (Mani et al., 2006) perform a 6-way classification, and not a 13-way classification[11], and (ii) (Chambers and Jurafsky, 2008) use a relation set that is even more restrictive than TempEval's.

6.3 Saturated graphs

Table 3 summarizes the various precision/recall scores of the graph obtained by saturating the classifiers predictions (potentially altered by NRO) against the event-event saturated graph. These results contrast with the accuracy results presented in table 2: while the TempEval-based model was the best model in classification accuracy in TempEval, it is now outperformed by both the Allen- and Bruce-based systems (this with or with using NRO). The best system in TempEval is actually Bruce-based system, with 52.9 and 62.8 for the strict/relaxed metrics, respectively. The results suggest that this algebra might actually offer the best trade-off between learnanility and expressive power. The use of NRO to restore global coherence yields important gains (10 points) in the relaxed metric for both Allen- and Bruce-based systems (although they do not convert into gains in the strict metric). Unsuprisingly, the best model on the Allen set remains Allen-based model (and this time the use of NRO results in gains on the strict metric). Predictions without

[11]This is only possible because they order the event-event pairs before submitting them to the classifier.

System	Allen						Tempeval					
	RELAX			STRICT			RELAX			STRICT		
	R	P	F1	R	P	F1	R	P	F1	R	P	F1
Allen	57.5	46.7	**51.5**	49.6	56.2	**52.7**	62.0	50.3	55.5	50.4	57.1	53.6
Bruce	46.0	39.0	42.1	18.0	44.0	25.9	62.9	52.6	**57.3**	50.9	57.0	**53.8**
Tempeval	37.1	35.9	36.5	14.0	44.0	21.2	49.3	47.1	48.2	21.7	44.2	29.1
Allen$_{NRO}$	44.8	60.1	**51.3**	57.2	62.9	**59.9**	63.8	67.0	65.3	45.2	60.6	51.8
Bruce$_{NRO}$	46.3	53.1	49.5	13.9	45.3	21.2	65.5	71.8	**68.5**	46.6	61.1	**52.9**
Tempeval$_{NRO}$	37.1	35.9	36.5	13.9	44.3	21.2	49.3	47.1	48.2	21.7	44.2	29.1

Table 3: Comparing Allen-, Bruce-, Tempeval-based classifiers saturated predictions on saturated event-event graph. The $_{NRO}$ subscript indicates whether the system uses NRO or not. Evaluation are given with respect to both Allen and Tempeval relation sets.

NRO yielded between 7.5 and 9% of inconsistent saturated graphs that were ignored by the evaluation, which means this impacted recall measures only.

7 Related work

Early work on temporal ordering (Passonneau, 1988; Webber, 1988; Lascarides and Asher, 1993) concentrated on studying the knowledge sources at play (such as tense, aspect, lexical semantics, rhetorical relations). The development of annotated resources like the TimeBank corpus (Pustejovsky et al., 2003) has triggered the development of machine learning systems (Mani et al., 2006; Tatu and Srikanth, 2008; Chambers and Jurafsky, 2008).

More recent work uses automatic classification methods, based on the TimeBank and Acquaint corpus, either as is, with inferential enrichment for training (Mani et al., 2006; Chambers et al., 2007), or supplied with the corrections of (Bethard et al., 2007), or are restricted to selected contexts, such as intra-sentential event relations (Li et al., 2004; Lapata and Lascarides, 2006). All of these assume that event pairs are preselected, so the task is only to determine what is the most likely relation between them. The best scores are obtained with the added assumption that the event-event pair can be pre-ordered (thus reducing the number of possible labels by 2).

More recently, (Bramsen et al., 2006) and subsequently (Chambers and Jurafsky, 2008) propose to use an Integer Linear Programming solver to enforce the consistency of a network of constraints while maximizing the score of local classification decisions. But these are restricted to the relations BEFORE and AFTER, which have very strong inference properties that cannot be generalised to other relations. The ILP strategy is not likely to scale up very well for richer relation sets, for the number of possible relations between two events (and thus the number of variables to put in the LP solver for each pair) is the order of $2^{|R|}$ (where R is the relation set), and each transitivity constraints generates an enormous amount of constraints.

8 Conclusion

We have investigated the role played by ontological choices in temporal representations by comparing three algebras with different granularities of relations and inferential powers. Our experiments on the Timebank/AQUAINT reveal that the TempEval relation set provides the best overall classification accuracy, but it provides much less informative temporal structures, and it does not provide enough inferences for being useful for enforcing consistency. By contrast, the other two relation sets are significantly harder to learn, but provide more richer inferences and are therefore more useful when global consistency is important. Bruce's 7 relations-based model appears to perform best in the TempEval evaluation, suggesting that this algebra provides the best trade-off between learnability and expressive power.

References

Allen, James. 1983. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, pages 832–843.

Berger, A., S. Della Pietra, and V. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

Bethard, Steven, James H. Martin, and Sara Klingenstein. 2007. Timelines from text: Identification of syntactic temporal relations. In *International Conference on Semantic Computing*, pages 11–18, Los Alamitos, CA, USA. IEEE Computer Society.

Boguraev, Branimir and Rie Ando. 2005. TimeML-compliant text analysis for temporal reasoning. In Kaelbling, Leslie Pack and Fausto Giunchiglia, editors, *Proceedings of IJCAI05*, pages 997–1003.

Bramsen, Philip, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. 2006. Inducing temporal graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 189–198, Sydney, Australia, July. Association for Computational Linguistics.

Bruce, B. 1972. A model for temporal references and its application in a question answering program. *Artificial Intelligence*, 3(1-3):1–25.

Chambers, Nathanael and Daniel Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 698–706, Honolulu, Hawaii, October. Association for Computational Linguistics.

Chambers, Nathanael, Shan Wang, and Daniel Jurafsky. 2007. Classifying temporal relations between events. In *ACL*. The Association for Computer Linguistics.

Filatova, Elena and Eduard Hovy. 2001. Assigning time-stamps to event-clauses. In Mani, I., J. Pustejovsky, and R Gaizauskas, editors, *The Language of Time: A Reader*. Oxford University Press.

Lapata, Maria and Alex Lascarides. 2006. Learning sentence-internal temporal relations. *J. Artif. Intell. Res. (JAIR)*, 27:85–117.

Lascarides, Alex and Nicholas Asher. 1993. Temporal interpretation, discourse relations and common sense entailment. *Linguistics and Philosophy*, 16:437–493.

Li, Wenjie, Kam-Fai Wong, Guihong Cao, and Chunfa Yuan. 2004. Applying machine learning to chinese temporal relation resolution. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 582–588, Barcelona, Spain, July.

Malouf, Robert. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *Proceedings of the Sixth Workshop on Natural Language Learning*, pages 49–55, Taipei, Taiwan.

Mani, Inderjeet, Marc Verhagen, Ben Wellner, Chong Min Lee, and James Pustejovsky. 2006. Machine learning of temporal relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 753–760, Sydney, Australia, July. Association for Computational Linguistics.

Passonneau, Rebecca J. 1988. A computational model of the semantics of tense and aspect. *Computational Linguistics*, 14(2):44–60.

Pustejovsky, James, Patrick Hanks, Roser Saurí, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, and Marcia Lazo. 2003. The TIMEBANK Corpus. In *Proceedings of Corpus Linguistics*, pages 647–656, Lancaster University, UK, March.

Setzer, Andrea, Robert Gaizauskas, and Mark Hepple. 2006. The Role of Inference in the Temporal Annotation and Analysis of Text. *Language Resources and Evaluation*, 39:243–265.

Tatu, Marta and Munirathnam Srikanth. 2008. Experiments with reasoning for temporal relations between events. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 857–864, Manchester, UK, August. Coling 2008 Organizing Committee.

Verhagen, Marc, Robert Gaizauskas, Franck Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. SemEval-2007 - 15: TempEval Temporal Relation Identification. In *Proceedings of SemEval workshop at ACL 2007*, Prague, Czech Republic, June. Association for Computational Linguistics, Morristown, NJ, USA.

Webber, Bonnie Lynn. 1988. Tense as discourse anaphor. *Computational Linguistics*, 14(2):61–73.

Generating Learner-Like Morphological Errors in Russian

Markus Dickinson
Indiana University
md7@indiana.edu

Abstract

To speed up the process of categorizing learner errors and obtaining data for languages which lack error-annotated data, we describe a linguistically-informed method for generating learner-like morphological errors, focusing on Russian. We outline a procedure to select likely errors, relying on guiding stem and suffix combinations from a segmented lexicon to match particular error categories and relying on grammatical information from the original context.

1 Introduction

Work on detecting grammatical errors in the language of non-native speakers covers a range of errors, but it has largely focused on syntax in a small number of languages (e.g., Vandeventer Faltin, 2003; Tetreault and Chodorow, 2008). In more morphologically-rich languages, learners naturally make many errors in morphology (Dickinson and Herring, 2008). Yet for many languages, there is a major bottleneck in system development: there are not enough error-annotated learner corpora which can be mined to discover the nature of learner errors, let alone enough data to train or evaluate a system. Our perspective is that one can speed up the process of determining the nature of learner errors via semi-automatic means, by generating plausible errors.

We set out to generate linguistically-plausible morphological errors for Russian, a language with rich inflections. Generating learner-like errors has practical and theoretical benefits. First, there is the issue of obtaining training data; as Foster and Andersen (2009) state, "The ideal situation for a grammatical error detection system is one where a large amount of labelled positive *and* negative evidence is available." Generated errors can bridge this gap by creating realistic negative evidence (see also Rozovskaya and Roth, 2010). As for evaluation data, generated errors have at least one advantage over real errors, in that we know precisely what the correct form is supposed to be, a problem for real learner data (e.g., Boyd, 2010).

By starting with a coarse error taxonomy, generating errors can improve categorization. Generated errors provide data for an expert—e.g., a language teacher—to search through, expanding the taxonomy with new error types or subtypes and/or deprecating error types which are unlikely. Given the lack of real learner data, this has the potential to speed up error categorization and subsequent system development. Furthermore, error generation techniques can be re-used, adjusting the errors for different learner levels, first languages, and so forth.

The error generation process can benefit by using linguistic properties to mimic learner variations. This can lead to more realistic errors, a benefit for machine learning (Foster and Andersen, 2009), and can also provide feedback for the linguistic representation used to generate errors by, e.g., demonstrating under which linguistic conditions certain error types are generated and under which they are not.

We are specifically interested in generating Russian morphological errors. To do this, we need a knowledge base representing Russian morphology, allowing us to manipulate linguistic properties. After outlining the coarse error taxonomy

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 259–267,
Beijing, August 2010

(section 2), we discuss enriching a part-of-speech (POS) tagger lexicon with segmentation information (section 3). We then describe the steps in error generation (section 4), highlighting decisions which provide insight for the analysis of learner language, and show the impact on POS tagging in section 5.

2 Error taxonomy

Russian is an inflecting language with relatively free word order, meaning that morphological syntactic properties are often encoded by affixes. In (1a), for example, the verb начина needs a suffix to indicate person and number, and ет is the third person singular form.[1] By contrast, (1b) illustrates a paradigm error: the suffix ит is third singular, but not the correct one. Generating such a form requires having access to individual morphemes and their linguistic properties.

(1) a. начина+ет [*nachina+et*]
begin-3s

b. *начина+ит [*nachina+it*]
begin-3s (diff. verb paradigm)

This error is categorized as a suffix error in figure 1, expanding the taxonomy in Dickinson and Herring (2008). Stem errors are similarly categorized, with *Semantic errors* defined with respect to a particular context (e.g., using a different stem than required by an activity).

For formation errors (#3), one needs to know how stems relate. For instance, some verbs change their form depending on the suffix, as in (2). In (2c), the stem and suffix are morphologically compatible, just not a valid combination. One needs to know that мож is a variant of мог.

(2) a. мог+ут [*mog+ut*]
can-3p

b. мож+ет [*mozh+et*]
can-3s

c. *мож+ут [*mozh+ut*] (#3)
can-3p (wrong formation)

Using a basic lexicon without such knowledge, it is hard to tell formation errors apart from lexicon incompleteness (see section 4.2.2). If можут (2c) is generated and is not in the lexicon, we do not know whether it is misformed or simply unattested. In this paper, we group together such cases, since this allows for a simpler and more quickly-derivable lexicon.

[1]For examples, we write the Cyrillic form and include a Roman transliteration (SEV 1362-78) for ease of reading.

0. Correct: The word is well-formed.
1. Stem errors:
   (a) Stem spelling error
   (b) Semantic error
2. Suffix errors:
   (a) Suffix spelling error
   (b) Lexicon error:
      i. Derivation error: The wrong POS is used (e.g., a noun as a verb).
      ii. Inherency error: The ending is for a different subclass (e.g., inanimate as an animate noun).
   (c) Paradigm error: The ending is from the wrong paradigm.
3. Formation errors: The stem does not follow appropriate spelling/sound change rules.
4. Syntactic errors: The form is correct, but used in an in appropriate syntactic context (e.g., nominative case in a dative context)

- Lexicon incompleteness: The form may be possible, but is not attested.

Figure 1: Error taxonomy

We have added syntactic errors, whereas Dickinson and Herring (2008) focused on strictly morphological errors. Learners make syntactic errors (e.g., Rubinstein, 1995; Rosengrant, 1987), and when creating errors, a well-formed word may result. In the future, syntactic errors can be subdivided (Boyd, 2010).

This classification is of *possible* errors, making no claim about the *actual* distribution of learner errors, and does not delve into issues such as errors stemming from first language interference (Rubinstein, 1995). Generating errors from the possible types allows one to investigate which types are plausible in which contexts.

It should be noted that we focus on inflectional morphology in Russian, meaning that we focus on suffixes. Prefixes are rarely used in Russian as inflectional markers; for example, prefixes mark semantically-relevant properties for verbs of motion. The choice of prefix is thus related to the overall word choice, an issue discussed under *Random stem generation* in section 4.2.4.

3 Enriching a POS lexicon

To create errors, we need a segmented lexicon with morphological information, as in (3). Here, the word могу (*mogu*, 'I am able to') is split into stem and suffix, with corresponding POS tags.[2]

(3) a. мог,Vm-----a-p,у,Vmip1s-a-p
 b. мож,Vm-----a-p,ет,Vmip3s-a-p
 c. мог,Vm-----a-p,NULL,Vmis-sma-p

The freely-available POS lexicon from Sharoff et al. (2008), specifically the file for the POS tagger TnT (Brants, 2000), contains full words (239,889 unique forms), with frequency information. Working with such a rich database, we only need segmentation, providing a quickly-obtained lexicon (cf. five years for a German lexicon in Geyken and Hanneforth, 2005).

In the future, one could switch to a different tagset, such as that in Hana and Feldman (2010), which includes reflexivity, animacy, and aspect features. One could also expand the lexicon, by adapting algorithms for analyzing unknown words (e.g., Mikheev, 1997), as suggested by Feldman and Hana (2010). Still, our lexicon continues the trend of linking traditional categories used for tagging with deeper analyses (Sharoff et al., 2008; Hana and Feldman, 2010).[3]

3.1 Finding segments/morphemes

We use a set of hand-crafted rules to segment words into morphemes, of the form: if the tag is x and the word ends with y, make y the suffix. Such rules are easily and quickly derivable from a textbook listing of paradigms. For certain exceptional cases, we write word-specific rules. Additionally, we remove word, tag pairs indicating punctuation or non-words (`PUNC`, `SENT`, `-`).

One could use a sophisticated method for lemmatizing words (e.g., Chew et al., 2008; Schone and Jurafsky, 2001), but we would likely have to clean the lexicon later; as Feldman and Hana (2010) point out, it is difficult to automatically guess the entries for a word, without POS information. Essentially, we write precise rules to specify part of the Russian system of suffixes; the lexicon then provides the stems for free.

We use the lexicon for generating errors, but it should be compatible with analysis. Thus, we focus on suffixes for beginning and intermediate learners. We can easily prune or add to the rule set later. From an analysis perspective, we need to specify that certain grammatical properties are in a tag (see below), as an analyzer is to support the provision of feedback. Since the rules are freely available,[4] changing these criteria for other purposes is straightforward.

3.1.1 Segmentation rules

We have written 1112 general morphology rules and 59 rules for the numerals 'one' through 'four,' based on the *Nachalo* textbooks (Ervin et al., 1997). A rule is simply a tag, suffix pair. For example, in (4), `Ncmsay` (<u>N</u>oun, <u>c</u>ommon, <u>m</u>asculine, <u>s</u>ingular, <u>a</u>ccusative, animate [<u>y</u>es]) words should end in either а (*a*) or я (*ya*).

(4) a. Ncmsay, а
 b. Ncmsay, я

A program consults this list and segments a word appropriately, requiring at least one character in the stem. In the case where multiple suffixes match (e.g., ени (*eni*) and и (*i*) for singular neuter locative nouns), the longer one is chosen, as it is unambiguously correct.

We add information in 101 of the 1112 rules. All numerals, for instance, are tagged as `Mc-s` (Nu<u>m</u>eral, <u>c</u>ardinal, [unspecified gender], <u>s</u>ingular). The tagset in theory includes properties such as case; they just were not marked (see footnote 6, though). Based on the ending, we add all

[2] POS tags are from the compositional tagset in Sharoff et al. (2008). A full description is at: `http://corpus.leeds.ac.uk/mocky/msd-ru.html`.

[3] This lexicon now includes lemma information, but each word is not segmented (Erjavec, 2010).

[4] `http://cl.indiana.edu/~boltundevelopment/`

possible analyses. Using an optional output tag, in (5), Mc-s could be genitive (g), locative (l), or dative (d) when it ends in и (*i*). These rules increase ambiguity, but are necessary for learner feedback.

(5) a. Mc-s, и, Mc-sg
  b. Mc-s, и, Mc-sl
  c. Mc-s, и, Mc-sd

In applying the rules, we generate stem tags, encoding properties constant across suffixes. Based on the word's tag (e.g., Ncmsay, cf. (4)) a stem is given a more basic tag (e.g., Ncm--y).

3.2 Lexicon statistics

To be flexible for future use, we have only enriched 90% of the words (248,014), removing every 10th word. Using the set of 1112 rules results in a lexicon with 190,450 analyses, where *analyses* are as in (3). For these 190,450 analyses, there are 117 suffix forms (e.g., я, *ya*) corresponding to 808 suffix analyses (e.g., <я, Ncmsay>). On average 3.6 suffix tags are observed with each stem-tag pair, but 22.2 tags are compatible, indicating incomplete paradigms.

4 Generating errors

4.1 Basic procedure

Taking the morpheme-based lexicon, we generate errors by randomly combining morphemes into full forms. Such randomness must be constrained, taking into account what types of errors are likely to occur.

The procedure is given in figure 2 and detailed in the following sections. First, we use the contextually-determined POS tag to restrict the space of possibilities. Secondly, given that random combinations of a stem and a suffix can result in many unlikely errors, we guide the combinations, using a loose notion of likelihood to ensure that the errors fall into a reasonable distribution. After examining the generated errors, one could restrict the errors even further. Thirdly, we compare the stem and suffix to determine the possible types of errors. A full form may have several different interpretations, and thus, lastly, we select the best interpretation(s).

1. Determine POS properties of the word to be generated (section 4.2.1).
2. Generate a full-form, via *guided* random stem and suffix combination (section 4.2.4).
3. Determine possible error analyses for the full form (section 4.2.2).
4. Select the error type(s) from among multiple possible interpretations (section 4.2.3).

Figure 2: Error generation procedure

By trying to determine the best error type in step 4, the generation process can provide insight into error analysis. This is important, given that suffixes are highly ambiguous; for example, ой (*-oj*) has at least 6 different uses for adjectives. Analysis is not simply generation in reverse, though. Importantly, error generation relies upon the context POS tag for the *intended* form, for the whole process. To morphologically analyze the corrupted data, one has to POS tag *corrupted* forms (see section 5).

4.2 Corruption

We use a corpus of 5 million words automatically tagged by TnT (Brants, 2000) and freely available online (Sharoff et al., 2008).[5] Because we want to make linguistically-informed corruptions, we corrupt only the words we have information for, identifying the words in the corpus which are found in the lexicon with the appropriate POS tag.[6] We also select only words which have inflectional morphology: nouns, verbs, adjectives, pronouns, and numerals.[7]

4.2.1 Determining word properties (step 1)

We use the POS tag to restrict the properties of a word, regardless of how exactly we corrupt it. Either the stem and its tag or the suffix and its tag

[5] See http://corpus.leeds.ac.uk/mocky/.

[6] We downloaded the TnT lexicon in 2008, but the corpus in 2009; although no versions are listed on the website, there are some discrepancies in the tags used (e.g., numeral tags now have more information). To accommodate, we use a looser match for determining whether a tag is known, namely checking whether the tags are compatible. In the future, one can tweak the rules to match the newer lexicon.

[7] Adverbs inflect for comparative forms, but we do not consider them here.

can be used as an invariant, to guide the generated form (section 4.2.4). In (6a), for instance, the adjective (`Af`) stem or plural instrumental suffix (`Afp-pif`) can be used as the basis for generation.

(6) a. Original: серыми (*serymi*, 'gray') $\mapsto$ сер/Af+ыми/Afp-pif

b. Corrupted: сер+ой (*seroj*)

The error type is defined in terms of the original word's POS tag. For example, when we generate a correctly-formed word, as in (6b), it is a syntactic error if it does not match this POS tag.

4.2.2 Determining error types (step 3)

Before discussing word corruption in step 2 (section 4.2.4), we need to discuss how error types are determined (this section) and how to handle multiple possibilities (section 4.2.3), as these steps help guide step 2. After creating a corrupted word, we elucidate all possible interpretations in step 3 by comparing each suffix analysis with the stem. If the stem and suffix form a legitimate word (in the wrong context), it is a syntactic error. Incompatible features means a derivation or inherency error, depending upon which features are incompatible. If the features are compatible, but there is no attested form, it is either a paradigm error—if we know of a different suffix with the same grammatical features—or a formation/incompleteness issue, if not.

This is a crude morphological analyzer (cf. Dickinson and Herring, 2008), but bases its analyses on what is known about the invariant part of the original word. If we use ыми (*ymi*) from (6a) as an invariant, for instance, we know to treat it as a plural instrumental adjective ending, regardless of any other possible interpretations, because that is how it was used in this context.

4.2.3 Selecting the error type (step 4)

Corrupted forms may have many possible analyses. For example, in (6b), the suffix ой (*oj*) has been randomly attached to the stem сер (*ser*). With the stem fixed as an adjective, the suffix could be a feminine locative adjective (syntactic error), a masculine nominative adjective (paradigm error), or an instrumental feminine noun (derivation error). Given what learners are likely to do, we can use some heuristics to restrict the set of possible error types.

First, we hypothesize that a correctly-formed word is more likely a correct form than a misformed word. This means that correct words and syntactic errors—correctly-formed words in the wrong context—have priority over other error types. For (6b), for instance, the syntactic error outranks the paradigm and derivation errors.

Secondly, we hypothesize that a contextually-appropriate word, even if misformed, is more likely the correct interpretation than a contextually-inappropriate word. When we have cases where there is: a) a correctly-formed word not matching the context (a syntactic error), and b) a malformed word which matches the context (e.g., a paradigm error), we list both possibilities.

Finally, derivation errors seem less likely than the others (a point confirmed by native speakers), giving them lower priority. Given these heuristics, not only can we rule out error types after generating new forms, but we can also split the error generation process into different steps.

4.2.4 Corrupting selected words (step 2)

Using these heuristics, we take a known word and generate errors based on a series of choices. For each choice, we randomly generate a number between 0 and 1 and choose based on a given threshold. Thresholds should be reset when more is known about error frequency, and more decisions added as error subtypes are added.

Decision #1: Correct forms The first choice is whether to corrupt the word or not. Currently, the threshold is set at 0.5. If we corrupt the word, we continue on to the next decision.

Decision #2: Syntactic errors We can either generate a syntactic or a morphological error. On the assumption that syntactic errors are more common, we currently set a threshold of 0.7, generating syntactic errors 70% of the time and morphological form errors 30% of the time.

To generate a correct form used incorrectly, we extract the stem from the word and randomly select a new suffix. We keep selecting a suffix until we obtain a valid form.[8] An example is given in (7): the original (7a) is a plural instrumental adjective, unspecified for gender; in (7b), it is singular nominative feminine.

(7) a. серыми глазами .
gray eyes .
Afp-**pi**f Ncmpin SENT

b. серая глазами .
Afp**fsn**f Ncmpin SENT

One might consider ensuring that each error differs from the original in only one property. Or one might want to co-vary errors, such that, in this case, the adjective and noun both change from instrumental to nominative. While this is easily accomplished algorithmically, we do not know whether learners obey these constraints. Generating errors in a relatively unbounded way can help pinpoint these types of constraints.

While the form in (7b) is unambiguous, syntactic errors can have more than one possible analysis. In (8), for instance, this word could be corrupted with an -ой (*-oj*) ending, indicating feminine singular genitive, instrumental, or locative. We include all possible forms.

(8) серой глазами .
Afpfsg.Afpfsi.Afpfsl Ncmpin SENT

Likewise, considering the heuristics in section 4.2.3, generating a syntactic error may lead to a form which may be contextually-appropriate. Consider (9): in (9a), the verb-preposition combination requires an accusative (`Ncns`**`a`**`n`). By changing -o to -e, we generate a form which could be locative case (`Ncns`**`l`**`n`, type #4) or, since -e can be an accusative marker, a misformed accusative with the incorrect paradigm (#2c). We list both possibilities.

(9) a. . . . смотрел в небо
. . . (he) looked into the sky
. . . Vmis-sma-p Sp-a Ncnsan

b. . . . в небе
. . . Sp-a Ncns**a**n+2c.Ncns**l**n+4

Syntactic errors obviously conflate many different error types. The taxonomy for German from Boyd (2010), for example, includes selection, agreement, and word order errors. Our syntactic errors are either selection (e.g., wrong case as object of preposition) or agreement errors (e.g., subject-verb disagreement in number). However, without accurate syntactic information, we cannot divvy up the error space as precisely. With the POS information, we can at least sort errors based on the ways in which they vary from the original (e.g., incorrect case).

Finally, if no syntactic error can be derived, we revert to the correct form. This happens when the lexicon contains only one form for a given stem. Without changing the stem, we cannot generate a new form which is verifiably correct.

Decision #3: Morphological errors The next decision is: should we generate a true morphological error or a spelling error? We currently bias this by setting a 0.9 threshold. The process for generating morphological errors (0.9) is described in the next few sections, after which spelling errors (0.1) are described. Surely, 10% is an underestimate of the amount of spelling errors (cf. Rosengrant, 1987); however, for refining a morphological error taxonomy, biasing towards morphological errors is appropriate.

Decision #4: Invariant morphemes When creating a context-dependent morphological error, we have to ask what the unit, or morpheme, is upon which the full form is dependent. The final choice is thus to select whether we keep the stem analysis constant and randomize the suffix or keep the suffix and randomize the stem. Consider that the stem is the locus of a word's semantic properties, and the (inflectional) suffix reflects syntactic properties. If we change the stem of a word, we completely change the semantics (error type #1b). Changing the suffix, on the other hand, creates a morphological error with the same basic semantics. We thus currently randomly generate a suffix 90% of the time.

Random suffix generation Randomly attaching a suffix to a fixed stem is the same procedure used above to generate syntactic errors. Here, however, we force the form to be incorrect, not allowing syntactic errors. If attaching a suffix re-

[8] We ensure that we do not generate the original form, so that the new form is contextually-inappropriate.

sults in a correct form (contextually-appropriate or not), we re-select a random suffix.

Similarly, the intention is to generate inherency (#2bii), paradigm (#2c), and formation (#3) errors (or lexicon incompleteness). All of these seem to be more likely than derivation (#2bi) errors, as discussed in section 4.2.3. If we allow any suffix to combine, we will overwhelmingly find derivation errors. As pointed out in Dickinson and Herring (2008), such errors can arise when a learner takes a Russian noun, e.g., душ (*dush*, 'shower') and attempts to use it as a verb, as in English, e.g., душу (*dushu*) with first person singular morphology. In such cases, we have the wrong stem being used with a contextually-appropriate ending. Derviation errors are thus best served with random stem selection, as described in the next section. To rule out derivation errors, we only keep suffix analyses which have the same major POS as the stem.

For some stems, particular types of errors are impossible to generate. a) Inherency errors do not occur for underspecified stems, as happens with adjectives. For example, нов- (*nov-*, 'new') is an adjective stem which is compatible with any adjective ending. b) Paradigm errors cannot occur for words whose suffixes in the lexicon have no alternate forms; for instance, there is only one way to realize a third singular nominative pronoun. c) Lexicon incompleteness cannot be posited for a word with a complete paradigm. These facts show that the generated error types are biased, depending upon the POS and the completeness of the lexicon.

Random stem generation Keeping the suffix fixed and randomly selecting a stem ties the generated form to the syntactic context, but changes the semantics. Thus, these generated errors are firstly semantic errors (#1b), featuring stems inappropriate for the context, in addition to having some other morphological error. The fact that, given a context, we have to generate two errors lends weight to the idea that these are less likely.

A randomly-generated stem will most likely be of a different POS class than the suffix, resulting in a derivation error (#2bi). Further, as with all morphological errors, we restrict the generated word not to be a correctly-formed word, and we do not allow the stem or the suffix to be closed class items. It makes little sense to put noun inflections on a preposition, for example, and derivation errors involve open class words.[9]

Spelling errors For spelling errors, we create an error simply by randomly inserting, deleting, or substituting a single character in the word.[10] This will either be a stem (#1a) or a suffix (#2a) error. It is worth noting that since we know the process of creating this error, we are able to compartmentalize spelling errors from morphological ones. An error analyzer, however, will have a harder time distinguishing them.

5 Tagging the corpus

Figure 3 presents the distribution of error types generated, where *Word* refers to the number of words with a particular error type, as opposed to the count of error type+*POS* pairs, as each word can have more than one POS for an error type (cf. (9b)). For the 780,924 corrupted words, there are 2.67 error type+POS pairs per corrupted word. Inherency (#2bii) errors in particular have many tags per word, since the same suffix can have multiple similar deviations from the original (cf. (8)). Figure 3 shows that we have generated roughly the distribution we wanted, based on our initial ideas of linguisic plausibility.

Type	Word	POS	Type	Word	POS
1a	19,661	19,661	1b-2bi	11,772	11,772
2a	6,560	6,560	1b-2bii	5,529	5,529
2bii	150,710	749,292	1b-2c	279	279
2c	94,211	94,211	1b-3+	1,770	1,770
4	524,269	721,051			
3+	83,763	208,208	1b-all	19,350	19,350

Figure 3: Distribution of generated errors

Without an error detection system, it is hard to gauge the impact of the error generation process. Although it is not a true evaluation of the error generation process, as a first step, we test a POS

[9]Learners often misuse, e.g., prepositions, but these errors do not affect morphology. Future work should examine the relation between word choice and derivation errors, including changes in prefixes.

[10]One could base spelling errors on known or assumed phonological confusions (cf. Hovermale and Martin, 2008).

tagger against the newly-created data. This helps test the difficulty of tagging corrupted forms, a needed step in the process of analyzing learner language. Note that for providing feedback, it seems desirable to have the POS tagger match the tag of the corrupted form. This is a different goal than developing POS taggers which are robust to noise (e.g., Bigert et al., 2003), where the tag should be of the original word.

To POS tag, we use the HMM tagger TnT (Brants, 2000) with the model from `http://corpus.leeds.ac.uk/mocky/`. The results on the generated data are in figure 4, using a lenient measure of accuracy: a POS tag is correct if it matches any of the tags for the hypothesized error types. The best performance is for uncorrupted known words,[11] but notable is that, out of the box, the tagger obtains 79% precision on corrupted words when compared to the generated tags, but is strongly divergent from the original (no longer correct) tags. Given that 67% ($\frac{524{,}269}{780{,}924}$) of words have a syntactic error—i.e., a well-formed word in the wrong context—this indicates that the tagger is likely relying on the form in the lexicon more than the context.

	Gold Tags		
	Original	Error	# words
Corrupted	3.8%	79.0%	780,924
Unchanged:			
Known	92.1%	92.1%	965,280
Unknown	81.9%	81.9%	3,484,909
Overall	72.1%	83.4%	5,231,113

Figure 4: POS tagging results, comparing tagger output to *Original* tags and *Error* tags

It is difficult to break down the results for corrupted words by error type, since many words are ambiguous between several different error types, and each interpretation may have a different POS tag. Still, we can say that words which are syntactic errors have the best tagging accuracy. Of the 524,269 words which may be syntactic errors, TnT matches a tag in 96.1% of cases. Suffix spelling errors are particularly in need of improvement: only 17.3% of these words are correctly tagged (compared to 62% for stem spelling errors). With an ill-formed suffix, the tagger simply does not have reliable information. To improve tagging for morphological errors, one should investigate which linguistic properties are being incorrectly tagged (cf. sub-tagging in Hana et al., 2004) and what roles distributional, morphological, or lexicon cues should play in tagging learner language (see also Díaz-Negrillo et al., 2010).

[11] *Known* here refers to being in the enriched lexicon, as these are the cases we specificaly did not corrupt.

6 Conclusions and Outlook

We have developed a general method for generating learner-like morphological errors, and we have demonstrated how to do this for Russian. While many insights are useful for doing error analysis (including our results for POS tagging the resulting corpus), generation proceeds from knowing grammatical properties of the original word. Generating errors based on linguistic properties has the potential to speed up the process of categorizing learner errors, in addition to creating realistic data for machine learning systems. As a side effect, we also added segmentation to a wide-coverage POS lexicon.

There are several directions to pursue. The most immediate step is to properly evaluate the quality of generated errors. Based on this analysis, one can refine the taxonomy of errors, and thereby generate even more realistic errors in a future iteration. Additionally, building from the initial POS tagging results, one can work on generally analyzing the morphology of learner language, including teasing apart what information a POS tagger needs to examine and dealing with multiple hypotheses (Dickinson and Herring, 2008).

Acknowledgements

I would like to thank Josh Herring, Anna Feldman, Jennifer Foster, and three anonymous reviewers for useful comments on this work.

References

Bigert, Johnny, Ola Knutsson and Jonas Sjöbergh (2003). Automatic Evaluation of Robustness and Degradation in Tagging and Parsing. In *Proceedings of RANLP-2003*. Borovets, Bulgaria, pp. 51–57.

Boyd, Adriane (2010). EAGLE: an Error-Annotated Corpus of Beginning Learner German. In *Proceedings of LREC-10*. Malta.

Brants, Thorsten (2000). TnT – A Statistical Part-of-Speech Tagger. In *Proceedings of ANLP-00*. Seattle, WA, pp. 224–231.

Chew, Peter A., Brett W. Bader and Ahmed Abdelali (2008). Latent Morpho-Semantic Analysis: Multilingual Information Retrieval with Character N-Grams and Mutual Information. In *Proceedings of Coling 2008*. Manchester, pp. 129–136.

Díaz-Negrillo, Ana, Detmar Meurers, Salvador Valera and Holger Wunsch (2010). Towards interlanguage POS annotation for effective learner corpora in SLA and FLT. *Language Forum* .

Dickinson, Markus and Joshua Herring (2008). Developing Online ICALL Exercises for Russian. In *The 3rd Workshop on Innovative Use of NLP for Building Educational Applications*. Columbus, OH, pp. 1–9.

Erjavec, Tomaž (2010). MULTEXT-East Version 4: Multilingual Morphosyntactic Specifications, Lexicons and Corpora. In *Proceedings of LREC-10*. Malta.

Ervin, Gerard L., Sophia Lubensky and Donald K. Jarvis (1997). *Nachalo: When in Russia . . .* . New York: McGraw-Hill.

Feldman, Anna and Jirka Hana (2010). *A Resource-light Approach to Morpho-syntactic Tagging*. Amsterdam: Rodopi.

Foster, Jennifer and Oistein Andersen (2009). GenERRate: Generating Errors for Use in Grammatical Error Detection. In *The 4th Workshop on Innovative Use of NLP for Building Educational Applications*. Boulder, CO, pp. 82–90.

Geyken, Alexander and Thomas Hanneforth (2005). TAGH: A Complete Morphology for German Based on Weighted Finite State Automata. In *FSMNLP 2005*. Springer, pp. 55–66.

Hana, Jirka and Anna Feldman (2010). A Positional Tagset for Russian. In *Proceedings of LREC-10*. Malta.

Hana, Jirka, Anna Feldman and Chris Brew (2004). A Resource-light Approach to Russian Morphology: Tagging Russian using Czech resources. In *Proceedings of EMNLP-04*. Barcelona, Spain.

Hovermale, DJ and Scott Martin (2008). Developing an Annotation Scheme for ELL Spelling Errors. In *Proceedings of MCLC-5 (Midwest Computational Linguistics Colloquium)*. East Lansing, MI.

Mikheev, Andrei (1997). Automatic Rule Induction for Unknown-Word Guessing. *Computational Linguistics* 23(3), 405–423.

Rosengrant, Sandra F. (1987). Error Patterns in Written Russian. *The Modern Language Journal* 71(2), 138–145.

Rozovskaya, Alla and Dan Roth (2010). Training Paradigms for Correcting Errors in Grammar and Usage. In *Proceedings of HLT-NAACL-10*. Los Angeles, California, pp. 154–162.

Rubinstein, George (1995). On Case Errors Made in Oral Speech by American Learners of Russian. *Slavic and East European Journal* 39(3), 408–429.

Schone, Patrick and Daniel Jurafsky (2001). Knowledge-Free Induction of Inflectional Morphologies. In *Proceedings of NAACL-01*. Pittsburgh, PA.

Sharoff, Serge, Mikhail Kopotev, Tomaž Erjavec, Anna Feldman and Dagmar Divjak (2008). Designing and evaluating Russian tagsets. In *Proceedings of LREC-08*. Marrakech.

Tetreault, Joel and Martin Chodorow (2008). The Ups and Downs of Preposition Error Detection in ESL Writing. In *Proceedings of COLING-08*. Manchester.

Vandeventer Faltin, Anne (2003). Syntactic error diagnosis in the context of computer assisted language learning. Thèse de doctorat, Université de Genève, Genève.

Resolving Object and Attribute Coreference in Opinion Mining

Xiaowen Ding
Department of Computer Science
University of Illinois at Chicago
dingxwsimon@gmail.com

Bing Liu
Department of Computer Science
University of Illinois at Chicago
liub@cs.uic.edu

Abstract

Coreference resolution is a classic NLP problem and has been studied extensively by many researchers. Most existing studies, however, are generic in the sense that they are not focused on any specific text. In the past few years, opinion mining became a popular topic of research because of a wide range of applications. However, limited work has been done on coreference resolution in opinionated text. In this paper, we deal with object and attribute coreference resolution. Such coreference resolutions are important because without solving it a great deal of opinion information will be lost, and opinions may be assigned to wrong entities. We show that some important features related to opinions can be exploited to perform the task more accurately. Experimental results using blog posts demonstrate the effectiveness of the technique.

1 Introduction

Opinion mining has been actively researched in recent years. Researchers have studied the problem at the document level (e.g., Pang et al., 2002; Tuney, 2002; Gamon et al., 2005) sentence and clause level (Wilson et al., 2004; Kim and Hovy, 2004), word level (e.g., Andreevskaia and Bergler, 2006; Hatzivassiloglou and McKeown, 1997; Esuli and Sebastiani, 2006; Kanayama and Nasukawa, 2006; Qiu et al., 2009), and attribute level (Hu and Liu 2004; Popescu and Etzioni, 2005; Ku et al., 2006; Mei et al., 2007; Titov and McDonald 2008). Here attributes mean different aspects of an object that has been commented on. Let us use the following example blog to illustrate the problem: "*I bought a Canon S500 camera yesterday. It looked beautiful. I took a few photos last night. They were amazing*". "It" in the second sentence refers to "*Canon S500 camera*", which is called an *object*. "They" in the fourth sentence refers to "photos", which is called an *attribute* of the object "*Canon S500 camera*". The usefulness of coreference resolution in this case is clear. Without resolving them, we lose opinions. That is, although we know that the second and fourth sentences express opinions, we do not know on what. Without knowing the opinion target, the opinion is of limited use. In (Nicolov et al., 2008), it was shown based on manually annotated data that opinion mining results can be improved by 10% if coreference resolution is used (the paper did not provide an algorithm).

In this paper, we propose the problem of *object and attribute coreference resolution* – the task of determining which mentions of objects and attributes refer to the same entities. Note that here *entities* refer to both objects and attributes, not the traditional named entities. To our knowledge, limited work has been done on this problem in the opinion mining context apart from a prior study on resolving opinion sources (or holders) (Stoyanov and Cardie 2006). Opinion sources or holders are the persons or organizations that hold some opinions on objects and attributes. In this paper, we do not deal with source resolution as we are mainly interested in opinion texts on the web, e.g., reviews, discussions and blogs. In such environments opinion sources are usually the authors of the posts, which are displayed in Web pages.

This work follows the *attribute-based opinion mining model* in (Hu and Liu 2004; Popescu and Etzioni, 2005). In their work, attributes are called *features*. We do not use the term "feature" in this paper to avoid confusion with the term "feature" used in machine learning.

Our primary interests in this paper are opi-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 268–276,
Beijing, August 2010

nions expressed on products and services, which are called *objects*. Each object is described by its parts/components and attributes, which are all called *attributes* for simplicity.

This paper takes the supervised learning approach to solving the problem. The key contribution of this paper is the design and testing of two novel opinion related features for learning. The first feature is based on sentiment analysis of normal sentences (non-comparative sentences), comparative sentences, and the idea of sentiment consistency. For example, we have the sentences, "*The Sony camera is better than the Canon camera. It is cheap too*." It is clear that "It" means "Sony" because in the first sentence, the opinion on "Sony" is positive (comparative positive), but negative (comparative negative) on "Canon", and the second sentence is positive. Thus, we can conclude that "It" refers to "Sony" because people usually express sentiments in a consistent way. It is unlikely that "It" refers to "Canon". This is the idea of *sentiment consistency*. As we can see, this feature requires the system to have the ability to determine positive and negative opinions expressed in normal and comparative sentences.

The second feature considers what objects and attributes are modified by what opinion words. Opinion words are words that are commonly used to express positive or negative opinions, e.g., good, best, bad, and poor. Consider the sentences, "*The picture quality of the Canon camera is very good. It is not expensive either*." The question is what "It" refers to, "Canon camera" or "picture quality". Clearly, we know that "It" refers to "Canon camera" because "picture quality" cannot be expensive. To make this feature work, we need to identify what opinion words are usually associated with what objects or attributes, which means that the system needs to discover such relationships from the corpus.

These two features give significant boost to the coreference resolution accuracy. Experimental results based on three corpora demonstrate the effectiveness of the proposed features.

2 Related Work

Coreference resolution is an extensively studied NLP problem (e.g., Morton, 2000; Ng and Cardie, 2002; Gasperin and Briscoe, 2008). Early knowledge-based approaches were domain and linguistic dependent (Carbonell and Brown 1988), where researchers focused on diverse lexical and grammatical properties of referring expressions (Soon et al., 2001; Ng and Cardie, 2002; Zhou et al., 2004). Recent research relied more on exploiting semantic information. For example, Yang et al. (2005) used the semantic compatibility information, and Yang and Su (2007) used automatically discovered patterns integrated with semantic relatedness information, while Ng (2007) employed semantic class knowledge acquired from the Penn Treebank. Versley et al. (2008) used several kernel functions in learning.

Perhaps, the most popular approach is based on supervised learning. In this approach, the system learns a pairwise function to predict whether a pair of noun phrases is coreferent. Subsequently, when making coreference resolution decisions on unseen documents, the learnt pairwise noun phrase coreference classifier is run, followed by a clustering step to produce the final clusters (coreference chains) of coreferent noun phrases. For both training and testing, coreference resolution algorithms rely on feature vectors for pairs of noun phrases that encode lexical, grammatical, and semantic information about the noun phrases and their local context.

Soon et al. (2001), for example, built a noun phrase coreference system based on decision trees and it was tested on two standard coreference resolution data sets (MUC-6, 1995; MUC-7, 1998), achieving performance comparable to the best-performing knowledge based coreference engines at that time. The learning algorithm used 12 surface-level features. Our proposed method builds on this system with additional sentiment related features. The features inherit from this paper includes:

Distance Feature: Its possible values are 0, 1, 2, 3 and so on which captures the sentence distance between two entities.

Antecedent-pronoun feature, anaphor-pronoun feature: If the candidate antecedent or anaphor is a pronoun, it is true; false otherwise.

Definite noun phrase feature: The value is true if the noun phrase starts with "the"; false otherwise.

Demonstrative noun phrase feature: The value is true if the noun phrase starts with the word "this", "that", "these", or "those"; false otherwise.

Number agreement feature: If the candidate antecedent and anaphor are both singular or both plural, the value is true; otherwise false.

Both-proper-name feature: If both the candidates are proper nouns, which are determined by capitalization, return true; otherwise false.

Alias feature: It is true if one candidate is an alias of the other or vice versa; false otherwise.

Ng and Cardie (2002) expanded the feature set of Soon et al. (2001) from 12 to 53 features. The system was further improved by Stoyanov and Cardie (2006) who gave a partially supervised clustering algorithm and tackled the problem of opinion source coreference resolution.

Centering theory is a linguistic approach tried to model the variation or shift of the main subject of the discourse in focus. In (Grosz et al., 1995; Tetreault, 2001), centering theory was applied to sort the antecedent candidates based on the ranking of the forward-looking centers, which consist of those discourse entities that can be interpreted by linguistic expressions in the sentences. Fang et al. (2009) employed the centering theory to replace the grammatical role features with semantic role information and showed superior accuracy performances.

Ding et al. (2009) studied the *entity assignment* problem. They tried to discover the product names discussed in forum posts and assign the product entities to each sentence. The work did not deal with product attributes.

Unsupervised approaches were also applied due to the cost of annotating large corpora. Ng (2008) used an Expectation-Maximization (EM) algorithm, and Poon and Domingos (2008) applied Markov Logic Network (MLN).

Another related work is the indirect anaphora, known as bridging reference. It arises when an entity is part of an earlier mention. Resolving indirect anaphora requires background knowledge (e.g. Fan et al., 2005), and it is thus not in the scope of this paper.

Our work differs from these existing studies as we work in the context of opinion mining, which gives us extra features to enable us to perform the task more effectively.

3 Problem of Object and Attribute Coreference Resolution

In general, opinions can be expressed on anything, e.g., a product, an individual, an organization, an event, a topic, etc. Following (Liu, 2006), we also use the term *object* to denote an named entity that has been commented on. The object has a set of *components* (or parts) and also a set of *attributes*. For simplicity, *attribute* is used to denote both component and attribute in this paper. Thus, we have the two concepts, *object* and *attribute*.

3.1 Objective

Task objective: To carry out coreference resolution on objects and attributes in opinion text.

As we discussed in the introduction section, coreference resolution on objects and attributes is important because they are the core entities on which people express opinions. Due to our objective, we do not evaluate other types of coreferences. We assume that objects and entities have been discovered by an existing system (e.g., Hu and Liu 2004, Popescu and Etzioni 2005). Recall that a coreference relation holds between two noun phrases if they refer to the same entity. For example, we have the following three consecutive sentences:

s_1: I love *the nokia n95* but not sure how good *the flash* would be?
s_2: and also *it* is quite expensive so anyone got any ideas?
s_3: I will be going on contract so as long as i can get a good deal of *it*.

"it" in s_2 refers to the entity "the nokia n95" in s_1. In this case, we call "the nokia n95" the *antecedent* and pronoun "it" in s_2 the *anaphor*. The referent of "it" in s_3 is also "the nokia n95", so the "it" in s_3 is coreferent with the "it" in s_2.

Our task is thus to decide which mentions of objects and attributes refer to the same entities.

3.2 Overview of Our Approach

Like traditional conference resolution, we employ the supervised learning approach by including additional new features. The main steps of our approach are as follows:

Preprocessing: We first preprocess the corpus by running a POS tagger[1], and a Noun Phrase finder[2]. We then produce the set O-NP which includes both possible objects, attributes and other noun phrases. The noun phrases are

[1] http://nlp.stanford.edu/software/tagger.shtml
[2] http://crfchunker.sourceforge.net/

found using the Noun Phrase finder and the object names are consecutive NNPs. O-NP thus contains everything that needs to be resolved.

Feature vector construction: To perform machine learning, we need a set of features. Similar to previous supervised learning approaches (Soon et al., 2001), a feature vector is formed for every pair of phrases in O-NP extracted in the preprocessing step. We use some of the features introduced by Soon et al. (2001) together with some novel new features that we propose in this work. Since our focus is on products and attributes in opinionated documents, we do not use personal pronouns, the gender agreement feature, and the appositive feature, as they are not essential in blogs and forum posts discussing products.

Classifier construction: Using the feature vectors obtained from the previous step, we construct the training data, which includes all pairs of manually tagged phrases that are either object names or attributes. More precisely, each pair contains at least one object or one attribute. Using the training data, a decision tree is constructed using WEKA[3].

Testing: The testing phase employs the same preprocessing and feature vector construction steps as described above, followed by the application of the learnt classifier on all candidate coreference pairs (which are represented as feature vectors). Since we are only interested in coreference information for objects and attribute noun phrases, we discard non-object and non-attribute noun phrases.

4 The Proposed New Features

On surface, object and attribute coreference resolution seems to be the same as the traditional noun phrase coreference resolution. We can apply an existing coreference resolution technique. However, as we mentioned earlier, in the opinion mining context, we can have a better solution by integrating opinion information into the traditional lexical and grammatical features. Below are several novel features that we have proposed. We use α_i to denote an antecedent candidate and α_j an anaphor candidate. Note that we will not repeat the features used in previous systems, but only focus on the new features.

[3] http://www.cs.waikato.ac.nz/ml/weka/

4.1 Sentiment Consistency

Intuitively, in a post, if the author starts expressing opinions on an object, he/she will continue to have the same opinion on that object or its attributes unless there are contrary words such as "but" and "however". For example, we have the following blog (an id is added before each sentence to facilitate later discussion):

> *"(1) I bought Camera-A yesterday. (2) I took a few pictures in the evening in my living room. (3) The images were very clear. (4) They were definitely better than those from my old Camera-B. (5a) It is cheap too. (5b) The pictures of that camera were blurring for night shots, but for day shots it was ok"*

The comparative sentence (4) says that Camera-A is superior to Camera-B. If the next sentence is (5a) ((5a) and (5b) are alternative sentences), "it" should refer to the superior product/object (Camera-A) because sentence (5a) expresses a positive opinion. Similarly, if the next sentence is sentence (5b) which expresses a negative opinion in its first clause, "that camera" should refer to the inferior product (Camera-B). We call this phenomenon *sentiment consistency* (*SC*), which says that consecutive sentiment expressions should be consistent with each other unless there are contrary words such as "but" and "however". It would be ambiguous if such consistency is not observed.

Following the above observation, we further observe that if the author wants to introduce a new object o, he/she has to state the name of the object explicitly in a sentence $s_{i\text{-}1}$. The question is what happens to the next sentence s_i if we need to resolve the pronouns in s_i.

We consider several cases:

1. $s_{i\text{-}1}$ is a normal sentence (not a comparative sentence). If s_i expresses a consistent sentiment with $s_{i\text{-}1}$, it should refer to the same object as $s_{i\text{-}1}$. For example, we have

   $s_{i\text{-}1}$: *The N73* is my favorite.

   s_i: *It* can produce great pictures.

   Here "It" in s_i clearly refers to "The N73" in the first sentence $s_{i\text{-}1}$.
2. $s_{i\text{-}1}$ is a normal sentence and s_i does not express a consistent sentiment, then α_i and α_j introduced in these two sentences may not be coreferenced. For example, we have

   $s_{i\text{-}1}$: *The K800* is awesome.

   s_i: *That phone* has short battery life.

Here "The K800" and "That phone" may not be a coreference pair according to sentiment consistency. "That phone" should refer to an object appeared in an earlier sentence.

3. $s_{i\text{-}1}$ is a comparative sentence. If s_i expresses a positive (respectively negative) sentiment, the pronoun in s_i should refer to the superior (or inferior) entity in $s_{i\text{-}1}$ to satisfy sentiment consistency. This situation is depicted in the earlier example blog. For completeness, we give another example.

   $s_{i\text{-}1}$: *The XBR4* is brighter than *the 5080*.

   s_i: Overall, *it* is a great choice.

   Here "it" in s_i should refer to "The XBR4" in $s_{i\text{-}1}$ since they both have positive sentiments expressed on them.

Opinion Mining of Comparative Sentences: To deal with case (3), we need to identify superior entities from comparative sentences. In fact, we first need to find such comparative sentences. There is a prior work on identifying comparative sentences (Jindal and Liu. 2006). Since our focus is not to identify such sentences, we used several heuristic rules based on some comparative keywords, e.g. *than*, *win*, *superior*, etc. They achieve the F-score of 0.9. We then followed the opinion mining method introduced in (Ding et al. 2009) to find superior entities. Since a comparative sentence typically has entities on the two sides of a comparative keyword, i.e., "*Camera-X is better than Camera-Y*", based on opinion mining, if the sentence is positive, then the entities before the comparative keyword is superior and otherwise they are inferior (with the negation considered).

SC Feature: The possible value for this feature is 0, 1, or 2. If α_i and α_j have the same opinion, return 1; different opinions, return 0; and if the opinions cannot be identified for one or both of them, return 2. Here is an example explaining how the feature is used in our system:

"*My wife has currently got a Nokia 7390, which is terrible. My 6233 would always get great reception, hers would get no signal*."

Using our algorithm for opinion mining, "hers" gets a negative opinion in the second sentence. So the value for this feature for the pair, "hers" and "a Nokia 7390", is 1. The feature value for the pair "hers" and "My 6233" is 0. The idea is that because the first sentence expresses a negative sentiment on "a Nokia 7390", and there is no discourse connective (such as "but" and "however") between these two sentences. "Hers" should be talking about "a Nokia 7390" so as to satisfy sentiment consistency.

4.2 Entity and Opinion Word Association

One of the most important factors determining the orientation of opinions is the opinion words that opinion holders use to express their opinions. Different entities may be modified by different opinion words. We can use their association information with entities (both objects and attributes) to identify their coreferences.

Opinion Words: In most cases, opinions in sentences are expressed using *opinion words*. For example, the sentence, "*The picture quality is amazing*", expresses a positive opinion on the "picture quality" attribute because of the positive opinion word "amazing".

Researchers have compiled sets of such words for adjectives, adverbs, verbs, and nouns respectively. Such lists are collectively called the *opinion lexicon*. We obtained an opinion lexicon from the authors of (Ding et al. 2009).

It is useful to note that opinion words used to express opinions on different entities are usually different apart from some general opinion words such as good, great, bad, etc, which can express opinions on almost anything. For example, we have the following passage:

"*i love the nokia n95 but not sure how strong the flash would be? And also it is quite expensive, so anyone got any ideas*?"

Here "strong" is an opinion word that expresses a positive opinion on "the flash", but is seldom used to describe "the nokia n95". "expensive", on the other hand, should not be associated with "the flash", but is an opinion word that indicates a negative opinion on "the nokia n95". So "the nokia n95" is more likely to be the antecedent of "it" in the second sentence.

The question is how to find such associations of entities and opinion words. We use their co-occurrence information to measure, i.e., the *pointwise mutual information* of the two terms. First, we estimate the probability of $P(NP)$, $P(OW)$ and $P(NP\&OW)$. Here *NP* means a noun phrase, e.g., an object (attribute) after removing determiners, and *OW* means an opinion word. To compute the probability, we first count the occurrences of the words. Then the probability is computed as follow:

$$P(NP, OW) = \frac{NumofS(NP \& OW)}{TotalNumofSentence}$$

where *NumofS* is a function that gives the number of sentences that contain the particular word string. *P*(*NP, OW*) is computed in the same way. Let us use the previous example again. We compute *P*("nokia n95","expensive") as the number of sentences containing both "nokia n95" and "expensive" divided by the total number of sentences in the whole corpus.

Then we use the pointwise mutual information between a noun phrase and an opinion word to measure the association.

$$PMI(NP, OW) = \log \frac{P(NP, OW)}{P(NP)P(OW)}$$

However, this PMI value cannot be encoded directly as a feature as it only captures the local information between antecedent candidates and opinion words. That is, it cannot be used as a global feature in the classifier. We thus rank all possible antecedents of anaphor α_j based on their *PMI* values and use the ranking as the feature value. The highest ranked antecedent α_i has value 1; the second one has value 2 and so on. The candidates ranked below the fourth place all have the value 5. In the example above, if *PMI*("nokia n95", "expensive") is greater than *PMI*("flash", "expensive"), the feature for "nokia n95" and "it" pair will have a smaller value than the feature for the "flash" and "it" pair.

One may ask if we can use all adjectives and adverbs to associate with objects and attributes rather than just opinion words since most opinion words are adjectives and adverbs. We tested that, but the results were poor. We believe the reason is that there are many adjectives and adverbs which are used for all kinds of purposes and may not be meaningful for our task.

4.3 String Similarity Feature

Soon et al. (2001) has a string match feature (SOON STR), which tests whether the two noun phrases are the same string after removing determiners from each. Ng and Cardie (2002) split this feature into several primitive features, depending on the type of noun phrases. They replace the SOON STR feature with three features — PRO STR, PN STR, and WORDS STR — which restrict the application of string matching to pronouns, proper names, and non-pronominal noun phrases, respectively.

In the user generated opinion data, these may not be sufficient. For a certain product, people can have a large number of ways to express it. For example, we have

"*Panasonic TH50PZ700U VS TH50PZ77U, Which Plasma tv should I go for. The TH77U is about $500.00 more than the 700U.*"

Here "TH77U" is the same entity as "Panasonic TH50PZ77U", and "TH50PZ700U" is the same as "700U". But they cannot be easily identified by "same string" features mentioned above. Although "700U" can be solved using substring features, "TH77U" is difficult to deal with.

We employ a modified edit distance to computing a similarity score between different mentions and use that as a feature in our system. When one candidate is a substring of another, return 1; otherwise, 1 plus the edit distance.

4.4 Other Useful Features

In the machine learning approach introduced by Soon et al. (2001), they had several general features that can deal with various kinds of entities, e.g., semantic class agreement features dealing with different semantic classes like date, location, etc., and the gender agreement feature related to personal entities. However, these features are not so useful for our task because the semantic class of a product in one domain is usually consistent, and dates and locations are unlikely to be of any products that people will express their opinions. Moreover, we do not study opinion holders (as they are known in the Web environment), so personal entities are not the aspect that we concentrate on. Thus we did not use the following features: semantic class agreement features, the gender agreement feature, and appositive feature.

However, we added some specific features, which are based on two extracted entities, α_i and α_j, where α_i is the potential antecedent and α_j is the potential anaphor:

Is-between feature: Its possible values are true and false. If the words between α_i and α_j have an is-like verb (i.e., is, are, was, were, and be) between them and there is no comparative indicators, this feature has the value of true, e.g., "*The nokia e65* is *a good handset*."

In sentences similar to this example, the entities before and after "is" usually refer to the same object or attribute by a definition relation.

And the value of this feature will be true.

If "is" appears together with a comparative word, it is probably an indication that the two entities are different, and the value for this feature will be false, e.g., "*Overall the K800 is far superior to the W810*."

Has-between feature: Its possible values are also true and false. If the words between α_i and α_j have a has-like verb (i.e., has, have, and had), the value is true, and otherwise false, e.g., "*The k800* has *a 3.2 megapixel camera*."

This feature usually indicates a "part-of" relation if "has" appears between two entities. They do not refer to the same entity. Table 1 gives a summary of all the features used in our system.

5 Experiments and Discussions

5.1 Datasets

For evaluation, we used forum discussions from three domains, mobile phones, plasma and LCD TVs, and cars. Table 2 shows the characteristics of the three data sets. Altogether, we downloaded 64 discussion threads, which contain 453 individual posts with a total of 3939 sentences. All the sentences and product names were annotated strictly following the MUC-7 coreference task annotation standard[4]. Here is an example:

"*Phil had <COREF ID = "6" TYPE = "OBJ">a z610</COREF> which has <COREF ID = "7" TYPE = "ATTR">a 2MP camera</COREF>, and he never had a problem with <COREF ID = "8" TYPE = "OBJ" REF = "6">it</COREF>*."

ID and REF features are used to indicate that there is a coreference link between two strings. ID is arbitrary but uniquely assigned to each noun phrase. REF uses the ID to indicate a coreference link. "TYPE" can be "OBJ" (an object or a product), or "ATTR" (an attribute of an object). The annotation was done by the first author and another student before the algorithm construction, and the annotated data sets will be made public for other researchers to use.

For our experiments, we used the J48-decision tree builder in WEKA, a popular of machine learning suite developed at the University of Waikato. We conducted 10-fold cross validation on each dataset.

[4] http://www-nlpir.nist.gov/related_projects/muc/proceedings/co_task.html

The performances are measured using the standard evaluation measures of precision (p), recall (r) and F-score (F), $F = 2pr/(p+r)$. As we stated in Section 3, we are only interested in object and attributes noun phrases. So in the testing phrases, we only compute the precision and recall based on those pairs of candidates that contain at least one object or attribute noun phrase in each pair. If both of the candidates are not an object or an attribute, we ignore them.

5.2 Baseline

As the baseline systems, we duplicated two representative systems. Baseline1 is the decision tree system in Soon et al. (2001). We do not use the semantic class agreement feature, gender agreement feature and appositive feature in the original 12 features for the reason discussed in Section 4.4. Thus, the total number of features in Baseline1 is 9. The second baseline (baseline2) is based on the centering theory from the semantic perspective introduced by Fang et al. (2009). Centering theory is a theory about the local discourse structure that models the interaction of referential continuity and the salience of discourse entities in the internal organization of a text. Fang et al. (2009) extended the centering theory from the grammar level to the semantic level in tracking the local discourse focus.

5.3 Results Analysis

Table 3 gives the experimental results of the two baseline systems and our system with different features included. From Table 3, we can make several observations.

(1) Comparing the results of Baseline1 and our system with all features (Our System (All)), the new features introduced in this paper improves Baseline1 on average by more than 9% in F-score.
(2) Comparing the results of Baseline2 and our system with all features (Our System (All)), our system performs better than Baseline2 by about 3 - 5%. We also observe that centering theory (Baseline2) is indeed better than the traditional decision tree.
(3) Our system with sentiment consistency (SC) makes a major difference. It improves Baseline1 (our method is based on Baseline1) by 5-6% in F-score.
(4) With the additional feature of entity and opinion association (EOA), the results are

Feature category	Feature	Remark
Opinion mining based features	Opinion consistency	1, if the opinion orientation of α_i is the same as α_j, 0 if the opinions are different, else 2
	Entity and opinion words association	1, 2, 3, 4, 5 which indicate the rank positive based on the *PMI* value introduced in Section 4.2
grammatical	i-Pronoun feature	1, if α_i is a pronoun, else 0
	j-Pronoun feature	1, if α_j is a pronoun, else 0
	Number agreement feature	1, if both of the noun phrases agree in numbers, else 0
	Definite feature	1, if α_j starts with the word "the", else 0
	Demonstrative feature	1, if α_j starts with the word "this", "that", "those", or "these", else 0
	Both proper-name feature	1, if α_i and α_j are both proper names, else 0
lexical	String similarity	The string similarity score between α_i and α_j
	Alias feature	1, If α_i is an alias of α_j or vice versa, else 0
Others	Distance feature	The sentence distance between the pair of noun phrases, 0 if they are in the same sentence
	Keywords between features	1, if some keywords exist between α_i and α_j, else 0. Details are discussed in Section 4.5

Table 1: Feature list: α_i denotes the antecedent candidate and α_j the anaphor candidate

	Posts	Sentences
Phone	168	1498
TVs	173	1376
Cars	112	1065
Total	453	3939

Table 2: Characteristics of the datasets

		Cellphone			TVs			Cars		
		p	r	**F**	p	r	**F**	p	r	**F**
1	Baseline1	0.66	0.57	0.**61**	0.67	0.61	0.**64**	0.70	0.63	0.**66**
2	Baseline2	0.70	0.64	0.**67**	0.72	0.65	0.**68**	0.76	0.70	0.**73**
3	Our System (SC)	0.71	0.64	0.**67**	0.73	0.66	0.**69**	0.74	0.69	0.**72**
4	Our System (SC+EOA)	0.74	0.68	0.**71**	0.74	0.68	0.**71**	0.77	0.71	0.**74**
5	Our System (All)	0.75	0.70	0.**72**	0.76	0.70	0.**73**	0.78	0.73	0.**75**

Table 3: Results of object and attribute coreference resolution

improved further by another 2-4%.

(5) Our system with all features (row 5) performs the best.

Paired *t*-tests were performed on the three systems, i.e., baseline1, baseline2, and our system (row 5). The tests show that the improvements of our method over both Baseline1 and Baseline2 are significant at the confidence level of 95% for the first two datasets. For the third dataset, the improvement over Baseline1 is also significant at the confidence level of 95%, while the improvement over Baseline2 is significant at the confidence level of 90%.

In summary, we can conclude that the new technique is effective and is markedly better than the existing methods. It is clear that the new features made a major difference.

6 Conclusion

This paper investigated the coreference resolution problem in the opinion mining context. In particular, it studied object and attribute resolutions which are crucial for improving opinion mining results. Although we still took the supervised learning approach, we proposed several novel features in the opinion mining context, e.g., sentiment consistency, and object/attribute and opinion word associations. Experimental results using forum posts demonstrated the effectiveness of the proposed technique. In our future work, we plan to further improve the method and discover some other opinion related features that can be exploited to produce more accurate results.

References

A. Andreevskaia and S. Bergler. 2006. *Mining WordNet for Fuzzy Sentiment: Sentiment Tag Extraction from WordNet Glosses*. EACL'06.

J. Carbonell and R. Brown. 1988. *Anaphora resolution: a multi-strategy approach. COLING'1988.*

G. Carenini, R. Ng, and A. Pauls. 2006. *Interactive multimedia summaries of evaluative text. IUI'06.*

X. Ding, B. Liu and L. Zhang. 2009. *Entity Discovery and Assignment for Opinion Mining Application.* KDD'09.

A. Esuli and F. Sebastiani. 2006. *Determining Term Subjectivity and Term Orientation for Opinion Mining*, EACL'06.

J. Fan, K. Barker and B. Porter. 2005. *Indirect anaphora resolution as semantic path search*. K-CAP'05.

C. Gasperin and T. Briscoe. 2008. *Statistical anaphora resolution in biomedical texts*. COLING'08

B. J. Grosz, A. K. Joshi and S. Weinstein. 1995. *Centering: a framework for modeling the local coherence of discourse*. Computational Linguistics, 21(2).

V. Hatzivassiloglou and K. McKeown. 1997. *Predicting the Semantic Orientation of Adjectives*. ACL-EACL'97.

M. Hu and B. Liu. 2004. *Mining and summarizing customer reviews*. KDD'04.

N. Jindal, and B. Liu. 2006. *Mining Comparative Sentences and Relations*. AAAI'06.

H. Kanayama and T. Nasukawa. 2006. *Fully Automatic Lexicon Expansion for Domain-Oriented Sentiment Analysis*. EMNLP'06.

S. Kim and E. Hovy. 2004. *Determining the Sentiment of Opinions*. COLING'04.

N. Kobayashi, K. Inui and Y. Matsumoto. 2007. *Extracting Aspect-Evaluation and Aspect-of Relations in Opinion Mining*. EMNLP-CoNLL'07.

F. Kong, G. Zhou, Q. Zhu and P. Qian. 2009. *Employing the Centering Theory in Pronoun Resolution from the Semantic Perspective.* EMNLP'09.

L.-W. Ku, Y.-T. Liang and H.-H. Chen. 2006. *Opinion Extraction, Summarization and Tracking in News and Blog Corpora*. CAAW'06.

B. Liu. 2006. *Web Data Mining*, Springer.

R. McDonald, K. Hannan, T Neylon, M. Wells, and J.Reynar. 2007. *Structured Models for Fine-to-Coarse Sentiment Analysis*. ACL-07.

Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai. 2007. *Topic Sentiment Mixture: Modeling Facets and Opinions in Weblogs*. WWW'07.

T. S. Morton. 2000. *Coreference for NLP applications*. ACL'00.

V. Ng and C. Cardie. 2002. *Improving machine learning approaches to coreference resolution*. ACL'02.

V. Ng. 2007. *Semantic Class Induction and Coreference Resolution*. ACL'07.

V. Ng. 2008. *Unsupervised Models for Coreference Resolution*. EMNLP'08.

N. Nicolov, F. Salvetti and S. *Ivanova, Sentiment analysis: Does coreference matter?* AISB'2008.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. *Thumbs up? Sentiment Classification Using Machine Learning Techniques*. EMNLP'02.

H. Poon and P. Domingos. 2008. *Joint Unsupervised Coreference Resolution with Markov Logic*. EMNLP'08, 650–659.

A-M. Popescu and O. Etzioni. 2005. *Extracting product features and opinions from reviews*. EMNLP'05.

Qiu, Guang, B. Liu, J. Bu and C. Chen. 2009 *Expanding Domain Sentiment Lexicon through Double Propagation*. IJCAI 2009.

W. M. Soon, H. T. Ng and D. Lim. 2001. *A machine learning approach to coreference resolution of noun phrase*. Computational Linguistics, 27(4).

V. Stoyanov, C. Cardie. 2006. *Partially supervised coreference resolution for opinion summarization through structured rule learning.* EMNLP'06.

I. Titov and R. McDonald. 2008, *A joint model of text and aspect ratings for sentiment summarization*, ACL'08.

J. Tetreault. 2001. *A corpus-based evaluation of centering and pronoun resolution*. Computational Linguistics. 27(4):507-520.

P. Turney. 2002. *Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews.* ACL'02.

Y. Versley, A. Moschitti, M. Poesio and X. Yang. 2008. *Coreference systems based on kernels methods.* COLING'08.

T. Wilson, J. Wiebe, and R. Hwa. 2004. *Just how mad are you? Finding strong and weak opinion clauses.* AAAI'04.

X. F. Yang, J. Su and C. L. Tan. 2005. *Improving Pronoun Resolution Using Statistics - Based Semantic Compatibility Information*. ACL'05.

X. F. Yang and J. Su. 2007. *Coreference Resolution Using Semantic Relatedness Information from Automatically Discovered Patterns.* ACL'07.

G. D. Zhou and J. Su. 2004. *A high-performance coreference resolution system using a multi-agent strategy*. COLING'04.

Entity Disambiguation for Knowledge Base Population

†**Mark Dredze** and †**Paul McNamee** and †**Delip Rao** and †**Adam Gerber** and ⋆**Tim Finin**
†Human Language Technology Center of Excellence, Center for Language and Speech Processing
Johns Hopkins University
⋆University of Maryland – Baltimore County
mdredze,mcnamee,delip,adam.gerber@jhu.edu, finin@umbc.edu

Abstract

The integration of facts derived from information extraction systems into existing knowledge bases requires a system to disambiguate entity mentions in the text. This is challenging due to issues such as non-uniform variations in entity names, mention ambiguity, and entities absent from a knowledge base. We present a state of the art system for entity disambiguation that not only addresses these challenges but also scales to knowledge bases with several million entries using very little resources. Further, our approach achieves performance of up to 95% on entities mentioned from newswire and 80% on a public test set that was designed to include challenging queries.

1 Introduction

The ability to identify entities like people, organizations and geographic locations (Tjong Kim Sang and De Meulder, 2003), extract their attributes (Pasca, 2008), and identify entity relations (Banko and Etzioni, 2008) is useful for several applications in natural language processing and knowledge acquisition tasks like populating structured knowledge bases (KB).

However, inserting extracted knowledge into a KB is fraught with challenges arising from natural language ambiguity, textual inconsistencies, and lack of world knowledge. To the discerning human eye, the "Bush" in "Mr. Bush left for the Zurich environment summit in Air Force One." is clearly the US president. Further context may reveal it to be the 43rd president, George W. Bush, and not the 41st president, George H. W. Bush. The ability to disambiguate a polysemous entity mention or infer that two orthographically different mentions are the same entity is crucial in updating an entity's KB record. This task has been variously called entity disambiguation, record linkage, or entity linking. When performed without a KB, entity disambiguation is called coreference resolution: entity mentions either within the same document or across multiple documents are clustered together, where each cluster corresponds to a single real world entity.

The emergence of large scale publicly available KBs like Wikipedia and DBPedia has spurred an interest in linking textual entity references to their entries in these public KBs. Bunescu and Pasca (2006) and Cucerzan (2007) presented important pioneering work in this area, but suffer from several limitations including Wikipedia specific dependencies, scale, and the assumption of a KB entry for each entity. In this work we introduce an entity disambiguation system for linking entities to corresponding Wikipedia pages designed for open domains, where a large percentage of entities will not be linkable. Further, our method and some of our features readily generalize to other curated KB. We adopt a supervised approach, where each of the possible entities contained within Wikipedia are scored for a match to the query entity. We also describe techniques to deal with large knowledge bases, like Wikipedia, which contain millions of entries. Furthermore, our system learns when to withhold a link when an entity has no matching KB entry, a task that has largely been neglected in prior research in cross-document entity coreference. Our system produces high quality predictions compared with recent work on this task.

2 Related Work

The information extraction oeuvre has a gamut of relation extraction methods for entities like persons, organizations, and locations, which can be classified as open- or closed-domain depending on the restrictions on extractable relations (Banko and Etzioni, 2008). Closed domain systems extract a fixed set of relations while in open-domain systems, the number and type of relations are unbounded. Extracted relations still require processing before they can populate a KB with facts: namely, entity linking and disambiguation.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 277–285,
Beijing, August 2010

Motivated by ambiguity in personal name search, Mann and Yarowsky (2003) disambiguate person names using biographic facts, like birth year, occupation and affiliation. When present in text, biographic facts extracted using regular expressions help disambiguation. More recently, the Web People Search Task (Artiles et al., 2008) clustered web pages for entity disambiguation.

The related task of cross document coreference resolution has been addressed by several researchers starting from Bagga and Baldwin (1998). Poesio et al. (2008) built a cross document coreference system using features from encyclopedic sources like Wikipedia. However, successful coreference resolution is insufficient for correct entity linking, as the coreference chain must still be correctly mapped to the proper KB entry.

Previous work by Bunescu and Pasca (2006) and Cucerzan (2007) aims to link entity mentions to their corresponding topic pages in Wikipedia but the authors differ in their approaches. Cucerzan uses heuristic rules and Wikipedia disambiguation markup to derive mappings from surface forms of entities to their Wikipedia entries. For each entity in Wikipedia, a context vector is derived as a prototype for the entity and these vectors are compared (via dot-product) with the context vectors of unknown entity mentions. His work assumes that all entities have a corresponding Wikipedia entry, but this assumption fails for a significant number of entities in news articles and even more for other genres, like blogs. Bunescu and Pasca on the other hand suggest a simple method to handle entities not in Wikipedia by learning a threshold to decide if the entity is not in Wikipedia. Both works mentioned rely on Wikipedia-specific annotations, such as category hierarchies and disambiguation links.

We just recently became aware of a system fielded by Li *et al.* at the TAC-KBP 2009 evaluation (2009). Their approach bears a number of similarities to ours; both systems create candidate sets and then rank possibilities using differing learning methods, but the principal difference is in our approach to NIL prediction. Where we simply consider absence (*i.e.,* the NIL candidate) as another entry to rank, and select the top-ranked option, they use a separate binary classifier to decide whether their top prediction is correct, or whether NIL should be output. We believe relying on features that are designed to inform whether absence is correct is the better alternative.

3 Entity Linking

We define *entity linking* as matching a textual entity mention, possibly identified by a named entity recognizer, to a KB entry, such as a Wikipedia page that is a canonical entry for that entity. An entity linking *query* is a request to link a textual entity mention in a given document to an entry in a KB. The system can either return a matching entry or `NIL` to indicate there is no matching entry. In this work we focus on linking organizations, geo-political entities and persons to a Wikipedia derived KB.

3.1 Key Issues

There are 3 challenges to entity linking:

Name Variations. An entity often has multiple mention forms, including abbreviations (Boston Symphony Orchestra vs. BSO), shortened forms (Osama Bin Laden vs. Bin Laden), alternate spellings (Osama vs. Ussamah vs. Oussama), and aliases (Osama Bin Laden vs. Sheikh Al-Mujahid). Entity linking must find an entry despite changes in the mention string.

Entity Ambiguity. A single mention, like Springfield, can match multiple KB entries, as many entity names, like people and organizations, tend to be polysemous.

Absence. Processing large text collections virtually guarantees that many entities will not appear in the KB (`NIL`), even for large KBs.

The combination of these challenges makes entity linking especially challenging. Consider an example of "William Clinton." Most readers will immediately think of the 42nd US president. However, the only two William Clintons in Wikipedia are "William de Clinton" the 1st Earl of Huntingdon, and "William Henry Clinton" the British general. The page for the 42nd US president is actually "Bill Clinton". An entity linking system must decide if either of the William Clintons are correct, even though neither are exact matches. If the system determines neither matches, should it return `NIL` or the variant "Bill Clinton"? If variants are acceptable, then perhaps "Clinton, Iowa" or "DeWitt Clinton" should be acceptable answers?

3.2 Contributions

We address these entity linking challenges.

Robust Candidate Selection. Our system is flexible enough to find name variants but sufficiently restrictive to produce a manageable candidate list despite a large-scale KB.

Features for Entity Disambiguation. We developed a rich and extensible set of features based on the entity mention, the source document, and the KB entry. We use a machine learning ranker to score each candidate.

Learning NILs. We modify the ranker to learn `NIL` predictions, which obviates hand tuning and importantly, admits use of additional features that are indicative of `NIL`.

Our contributions differ from previous efforts (Bunescu and Pasca, 2006; Cucerzan, 2007) in several important ways. First, previous efforts depend on Wikipedia markup for significant performance gains. We make no such assumptions, although we show that optional Wikipedia features lead to a slight improvement. Second, Cucerzan does not handle `NIL`s while Bunescu and Pasca address them by learning a threshold. Our approach *learns* to predict `NIL` in a more general and direct way. Third, we develop a rich feature set for entity linking that can work with any KB. Finally, we apply a novel finite state machine method for *learning* name variations. [1]

The remaining sections describe the candidate selection system, features and ranking, and our novel approach learning `NIL`s, followed by an empirical evaluation.

4 Candidate Selection for Name Variants

The first system component addresses the challenge of name variants. As the KB contains a large number of entries (818,000 entities, of which 35% are PER, ORG or GPE), we require an efficient selection of the relevant candidates for a query.

Previous approaches used Wikipedia markup for filtering – only using the top-k page categories (Bunescu and Pasca, 2006) – which is limited to Wikipedia and does not work for general KBs. We consider a KB independent approach to selection that also allows for tuning candidate set size. This involves a linear pass over KB entry names (Wikipedia page titles): a naive implementation took two minutes per query. The following section reduces this to under two *seconds* per query.

For a given query, the system selects KB entries using the following approach:

- Titles that are exact matches for the mention.
- Titles that are wholly contained in or contain the mention (e.g., *Nationwide* and *Nationwide Insurance*).
- The first letters of the entity mention match the KB entry title (e.g., *OA* and *Olympic Airlines*).
- The title matches a known alias for the entity (aliases described in Section 5.2).
- The title has a strong string similarity score with the entity mention. We include several measures of string similarity, including: character Dice score > 0.9, skip bigram Dice score > 0.6, and Hamming distance $<= 2$.

We did not optimize the thresholds for string similarity, but these could obviously be tuned to minimize the candidate sets and maximize recall.

All of the above features are general for any KB. However, since our evaluation used a KB derived from Wikipedia, we included a few Wikipedia specific features. We added an entry if its Wikipedia page appeared in the top 20 Google results for a query.

On the training dataset (Section 7) the selection system attained a recall of 98.8% and produced candidate lists that were three to four orders of magnitude smaller than the KB. Some recall errors were due to inexact acronyms: ABC (Arab Banking; 'Corporation' is missing), ASG (Abu Sayyaf; 'Group' is missing), and PCF (French Communist Party; French reverses the order of the pre-nominal adjectives). We also missed International Police (Interpol) and Becks (David Beckham; Mr. Beckham and his wife are collectively referred to as 'Posh and Becks').

[1] http://www.clsp.jhu.edu/ markus/fstrain

4.1 Scaling Candidate Selection

Our previously described candidate selection relied on a linear pass over the KB, but we seek more efficient methods. We observed that the above non-string similarity filters can be pre-computed and stored in an index, and that the skip bigram Dice score can be computed by indexing the skip bigrams for each KB title. We omitted the other string similarity scores, and collectively these changes enable us to avoid a linear pass over the KB. Finally we obtained speedups by serving the KB concurrently[2]. Recall was nearly identical to the full system described above: only two more queries failed. Additionally, more than 95% of the processing time was consumed by Dice score computation, which was only required to correctly retrieve less than 4% of the training queries. Omitting the Dice computation yielded results in a few milliseconds. A related approach is that of canopies for scaling clustering for large amounts of bibliographic citations (McCallum et al., 2000). In contrast, our setting focuses on alignment vs. clustering mentions, for which overlapping partitioning approaches like canopies are applicable.

5 Entity Linking as Ranking

We select a single correct candidate for a query using a supervised machine learning ranker. We represent each query by a D dimensional vector $\mathbf{x}$, where $\mathbf{x} \in \mathbb{R}^D$, and we aim to select a single KB entry y, where $y \in \mathcal{Y}$, a set of possible KB entries for this query produced by the selection system above, which ensures that $\mathcal{Y}$ is small. The ith query is given by the pair $\{\mathbf{x}_i, y_i\}$, where we assume at most one correct KB entry.

To evaluate each candidate KB entry in $\mathcal{Y}$ we create feature functions of the form $f(\mathbf{x}, y)$, dependent on both the example $\mathbf{x}$ (document and entity mention) and the KB entry y. The features address name variants and entity disambiguation.

We take a maximum margin approach to learning: the correct KB entry y should receive a higher score than all other possible KB entries $\hat{y} \in \mathcal{Y}, \hat{y} \neq y$ plus some margin γ. This learning constraint is equivalent to the ranking SVM algorithm of Joachims (2002), where we define an ordered pair constraint for each of the incorrect KB entries $\hat{y}$ and the correct entry y. Training sets parameters such that score$(y) \geq$ score$(\hat{y}) + \gamma$. We used the library SVM$^{\text{rank}}$ to solve this optimization problem.[3] We used a linear kernel, set the slack parameter C as 0.01 times the number of training examples, and take the loss function as the total number of swapped pairs summed over all training examples. While previous work used a custom kernel, we found a linear kernel just as effective with our features. This has the advantage of efficiency in both training and prediction [4] – important considerations in a system meant to scale to millions of KB entries.

5.1 Features for Entity Disambiguation

200 atomic features represent $\mathbf{x}$ based on each candidate query/KB pair. Since we used a linear kernel, we explicitly combined certain features (e.g., acroynym-match AND known-alias) to model correlations. This included combining each feature with the predicted type of the entity, allowing the algorithm to learn prediction functions specific to each entity type. With feature combinations, the total number of features grew to 26,569. The next sections provide an overview; for a detailed list see McNamee et al. (2009).

5.2 Features for Name Variants

Variation in entity name has long been recognized as a bane for information extraction systems. Poor handling of entity name variants results in low recall. We describe several features ranging from simple string match to finite state transducer matching.

String Equality. If the query name and KB entry name are identical, this is a strong indication of a match, and in our KB entry names are distinct. However, similar or identical entry names that refer to distinct entities are often qualified with parenthetical expressions or short clauses. As an example, “London, Kentucky” is distinguished

[2]Our Python implementation with indexing features and four threads achieved up to 80× speedup compared to naive implementation.

[3]www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

[4]Bunescu and Pasca (2006) report learning tens of thousands of support vectors with their “taxonomy” kernel while a linear kernel represents all support vectors with a single weight vector, enabling faster training and prediction.

from "London, Ontario", "London, Arkansas", "London (novel)", and "London". Therefore, other string equality features were used, such as whether names are equivalent after some transformation. For example, "Baltimore" and "Baltimore City" are exact matches after removing a common GPE word like city; "University of Vermont" and "University of VT" match if VT is expanded.

Approximate String Matching. Many entity mentions will not match full names exactly. We added features for character Dice, skip bigram Dice, and left and right Hamming distance scores. Features were set based on quantized scores. These were useful for detecting minor spelling variations or mistakes. Features were also added if the query was wholly contained in the entry name, or vice-versa, which was useful for handling ellipsis (e.g., "United States Department of Agriculture" vs. "Department of Agriculture"). We also included the ratio of the recursive longest common subsequence (Christen, 2006) to the shorter of the mention or entry name, which is effective at handling some deletions or word reorderings (e.g., "Li Gong" and "Gong Li"). Finally, we checked whether all of the letters of the query are found in the same order in the entry name (e.g., "Univ Wisconsin" would match "University of Wisconsin").

Acronyms. Features for acronyms, using dictionaries and partial character matches, enable matches between "MIT" and "Madras Institute of Technology" or "Ministry of Industry and Trade."

Aliases. Many aliases or nicknames are nontrivial to guess. For example JAVA is the stock symbol for Sun Microsystems, and "Ginger Spice" is a stage name of Geri Halliwell. A reasonable way to do this is to employ a dictionary and alias lists that are commonly available for many domains[5].

FST Name Matching. Another measure of surface similarity between a query and a candidate was computed by training finite-state transducers similar to those described in Dreyer et al. (2008). These transducers assign a score to any string pair by summing over all alignments and scoring all contained character n-grams; we used n-grams of length 3 and less. The scores are combined using a global log-linear model. Since different spellings of a name may vary considerably in length (e.g., *J Miller* vs. *Jennifer Miller*) we eliminated the limit on consecutive insertions used in previous applications.[6]

[5]We used multiple lists, including class-specific lists (i.e., for PER, ORG, and GPE) lists extracted from Freebase (Bollacker et al., 2008) and Wikipedia redirects. PER, ORG, and GPE are the commonly used terms for entity types for people, organizations and geo-political regions respectively.

5.3 Wikipedia Features

Most of our features do not depend on Wikipedia markup, but it is reasonable to include features from KB properties. Our feature ablation study shows that dropping these features causes a small but statistically significant performance drop.

WikiGraph statistics. We added features derived from the Wikipedia graph structure for an entry, like indegree of a node, outdegree of a node, and Wikipedia page length in bytes. These statistics favor common entity mentions over rare ones.

Wikitology. KB entries can be indexed with human or machine generated metadata consisting of keywords or categories in a domain-appropriate taxonomy. Using a system called *Wikitology*, Syed et al. (2008) investigated use of ontology terms obtained from the explicit category system in Wikipedia as well as relationships induced from the hyperlink graph between related Wikipedia pages. Following this approach we computed top-ranked categories for the query documents and used this information as features. If none of the candidate KB entries had corresponding highly-ranked Wikitology pages, we used this as a `NIL` feature (Section 6.1).

5.4 Popularity

Although it may be an unsafe bias to give preference to common entities, we find it helpful to provide estimates of entity popularity to our ranker as others have done (Fader et al., 2009). Apart from the graph-theoretic features derived from the Wikipedia graph, we used Google's PageRank to by adding features indicating the rank of the KB entry's corresponding Wikipedia page in a Google query for the target entity mention.

[6]Without such a limit, the objective function may diverge for certain parameters of the model; we detect such cases and learn to avoid them during training.

5.5 Document Features

The mention document and text associated with a KB entry contain context for resolving ambiguity.

Entity Mentions. Some features were based on presence of names in the text: whether the query appeared in the KB text and the entry name in the document. Additionally, we used a named-entity tagger and relation finder, SERIF (Boschee et al., 2005), identified name and nominal mentions that were deemed co-referent with the entity mention in the document, and tested whether these nouns were present in the KB text. Without the NE analysis, accuracy on non-NIL entities dropped 4.5%.

KB Facts. KB nodes contain infobox attributes (or facts); we tested whether the fact text was present in the query document, both locally to a mention, or anywhere in the text. Although these facts were derived from Wikipedia infoboxes, they could be obtained from other sources as well.

Document Similarity We measured similarity between the query document and the KB text in two ways: cosine similarity with TF/IDF weighting (Salton and McGill, 1983); and using the Dice coefficient over bags of words. IDF values were approximated using counts from the Google 5-gram dataset as by Klein and Nelson (2008).

Entity Types. Since the KB contained types for entries, we used these as features as well as the predicted NE type for the entity mention in the document text. Additionally, since only a small number of KB entries had PER, ORG, or GPE types, we also inferred types from Infobox class information to attain 87% coverage in the KB. This was helpful for discouraging selection of eponymous entries named after famous entities (e.g., the former U.S. president vs. "John F. Kennedy International Airport").

5.6 Feature Combinations

To take into account feature dependencies we created combination features by taking the cross-product of a small set of diverse features. The attributes used as combination features included entity type; a popularity based on Google's rankings; document comparison using TF/IDF; coverage of co-referential nouns in the KB node text; and name similarity. The combinations were cascaded to allow arbitrary feature conjunctions. Thus it is possible to end up with a feature *kbtype-is-ORG AND high-TFIDF-score AND low-name-similarity*. The combined features increased the number of features from roughly 200 to 26,000.

6 Predicting NIL Mentions

So far we have assumed that each example has a correct KB entry; however, when run over a large corpus, such as news articles, we expect a significant number of entities will not appear in the KB. Hence it will be useful to predict `NIL`s.

We *learn* when to predict `NIL` using the SVM ranker by augmenting $\mathcal{Y}$ to include `NIL`, which then has a single feature unique to `NIL` answers. It can be shown that (modulo slack variables) this is equivalent to learning a single threshold τ for `NIL` predictions as in Bunescu and Pasca (2006).

Incorporating `NIL` into the ranker has several advantages. First, the ranker can set the threshold optimally without hand tuning. Second, since the SVM scores are relative within a single example and cannot be compared across examples, setting a single threshold is difficult. Third, a threshold sets a uniform standard across all examples, whereas in practice we may have reasons to favor a `NIL` prediction in a given example. We design features for `NIL` prediction that cannot be captured in a single parameter.

6.1 NIL Features

Integrating `NIL` prediction into learning means we can define arbitrary features indicative of `NIL` predictions in the feature vector corresponding to `NIL`. For example, if many candidates have good name matches, it is likely that one of them is correct. Conversely, if no candidate has high entry-text/article similarity, or overlap between facts and the article text, it is likely that the entity is absent from the KB. We included several features, such as a) the max, mean, and difference between max and mean for 7 atomic features for all KB candidates considered, b) whether any of the candidate entries have matching names (exact and fuzzy string matching), c) whether any KB entry was a top Wikitology match, and d) if the top Google match was not a candidate.

	Micro-Averaged				Macro-Averaged			
	Best	*Median*	*All Features*	*Best Features*	*Best*	*Median*	*All Features*	*Best Features*
All	0.8217	0.7108	0.7984	0.7941	0.7704	0.6861	0.7695	**0.7704**
non-NIL	0.7725	0.6352	0.7063	0.6639	0.6696	0.5335	0.6097	0.5593
NIL	0.8919	0.7891	0.8677	**0.8919**	0.8789	0.7446	0.8464	0.8721

Table 1: Micro and macro-averaged accuracy for TAC-KBP data compared to best and median reported performance. Results are shown for all features as well as removing a small number of features using feature selection on development data.

7 Evaluation

We evaluated our system on two datasets: the Text Analysis Conference (TAC) track on Knowledge Base Population (TAC-KBP) (McNamee and Dang, 2009) and the newswire data used by Cucerzan (2007) (Microsoft News Data).

Since our approach relies on supervised learning, we begin by constructing our own training corpus.[7] We highlighted 1496 named entity mentions in news documents (from the TAC-KBP document collection) and linked these to entries in a KB derived from Wikipedia infoboxes. [8] We added to this collection 119 sample queries from the TAC-KBP data. The total of 1615 training examples included 539 (33.4%) PER, 618 (38.3%) ORG, and 458 (28.4%) GPE entity mentions. Of the training examples, 80.5% were found in the KB, matching 300 unique entities. This set has a higher number of NIL entities than did Bunescu and Pasca (2006) (10%) but lower than the TAC-KBP test set (43%).

All system development was done using a train (908 examples) and development (707 examples) split. The TAC-KBP and Microsoft News data sets were held out for final tests. A model trained on all 1615 examples was used for experiments.

7.1 TAC-KBP 2009 Experiments

The KB is derived from English Wikipedia pages that contained an infobox. Entries contain basic descriptions (article text) and attributes. The TAC-KBP query set contains 3904 entity mentions for 560 distinct entities; entity type was only provided for evaluation. The majority of queries were for organizations (69%). Most queries were missing from the KB (57%). 77% of the distinct GPEs in the queries were present in the KB, but for PERs and ORGs these percentages were significantly lower, 19% and 30% respectively.

Table 1 shows results on TAC-KBP data using all of our features as well a subset of features based on feature selection experiments on development data. We include scores for both micro-averaged accuracy – averaged over all queries – and macro-averaged accuracy – averaged over each unique entity – as well as the best and median reported results for these data (McNamee and Dang, 2009). We obtained the best reported results for macro-averaged accuracy, as well as the best results for NIL detection with micro-averaged accuracy, which shows the advantage of our approach to learning NIL. See McNamee et al. (2009) for additional experiments.

The candidate selection phase obtained a recall of 98.6%, similar to that of development data. Missed candidates included *Iron Lady*, which refers metaphorically to Yulia Tymoshenko, *PCC*, the Spanish-origin acronym for the Cuban Communist Party, and *Queen City*, a former nickname for the city of Seattle, Washington. The system returned a mean of 76 candidates per query, but the median was 15 and the maximum 2772 (*Texas*). In about 10% of cases there were four or fewer candidates and in 10% of cases there were more than 100 candidate KB nodes. We observed that ORGs were more difficult, due to the greater variation and complexity in their naming, and that they can be named after persons or locations.

7.2 Feature Effectiveness

We performed two feature analyses on the TAC-KBP data: an additive study – starting from a small baseline feature set used in candidate selection we add feature groups and measure performance changes (omitting feature combinations), and an ablative study – starting from all features, remove a feature group and measure performance.

[7]Data available from www.dredze.com

[8]http://en.wikipedia.org/wiki/Help:Infobox

Class	*All*	*non-NIL*	*NIL*
Baseline	0.7264	0.4621	0.9251
Acronyms	0.7316	0.4860	0.9161
NE Analysis	0.7661	0.7181	0.8022
Google	0.7597	0.7421	0.7730
Doc/KB Text Similarity	0.7313	0.6699	0.7775
Wikitology	0.7318	0.4549	0.9399
All	0.7984	0.7063	0.8677

Table 2: Additive analysis: micro-averaged accuracy.

	Num. Queries		Accuracy		
	Total	*Nil*	*All*	*non-NIL*	*NIL*
NIL	452	187	0.4137	0.0	1.0
GPE	132	20	0.9696	1.00	0.8000
ORG	115	45	0.8348	0.7286	1.00
PER	205	122	0.9951	0.9880	1.00
All	452	187	**0.9469**	0.9245	0.9786
Cucerzan (2007)			0.914	-	-

Table 3: Micro-average results for Microsoft data.

Table 2 shows the most significant features in the feature addition experiments. The baseline includes only features based on string similarity or aliases and is not effective at finding correct entries and strongly favors NIL predictions. Inclusion of features based on analysis of named-entities, popularity measures (e.g., Google rankings), and text comparisons provided the largest gains. The overall changes are fairly small, roughly $\pm 1\%$; however changes in non-NIL precision are larger.

The ablation study showed considerable redundancy across feature groupings. In several cases, performance could have been slightly improved by removing features. Removing all feature combinations would have improved overall performance to 81.05% by gaining on non-NIL for a small decline on NIL detection.

7.3 Experiments on Microsoft News Data

We downloaded the evaluation data used in Cucerzan (2007)[9]: 20 news stories from MSNBC with 642 entity mentions manually linked to Wikipedia and another 113 mentions not having any corresponding link to Wikipedia.[10] A significant percentage of queries were not of type PER, ORG, or GPE (e.g., "Christmas"). SERIF assigned entity types and we removed 297 queries not recognized as entities (counts in Table 3).

We learned a new model on the training data above using a reduced feature set to increase speed.[11] Using our fast candidate selection system, we resolved each query in 1.98 seconds (median). Query processing time was proportional to the number of candidates considered. We selected a median of 13 candidates for PER, 12 for ORG and 102 for GPE. Accuracy results are in Table 3. The high results reported for this dataset over TAC-KBP is primarily because we perform very well in predicting popular and rare entries – both of which are common in newswire text.

One issue with our KB was that it was derived from infoboxes in Wikipedia's Oct 2008 version which has both new entities, [12] and is missing entities.[13] Therefore, we manually confirmed NIL answers and new answers for queries marked as NIL in the data. While an exact comparison is not possible (as described above), our results (94.7%) appear to be at least on par with Cucerzan's system (91.4% overall accuracy).With the strong results on TAC-KBP, we believe that this is strong confirmation of the effectiveness of our approach.

8 Conclusion

We presented a state of the art system to disambiguate entity mentions in text and link them to a knowledge base. Unlike previous approaches, our approach readily ports to KBs other than Wikipedia. We described several important challenges in the entity linking task including handling variations in entity names, ambiguity in entity mentions, and missing entities in the KB, and we showed how to each of these can be addressed. We described a comprehensive feature set to accomplish this task in a supervised setting. Importantly, our method discriminately learns when not to link with high accuracy. To spur further research in these areas we are releasing our entity linking system.

[9] http://research.microsoft.com/en-us/um/people/silviu/WebAssistant/TestData/

[10] One of the MSNBC news articles is no longer available so we used 759 total entities.

[11] We removed Google, FST and conjunction features which reduced system accuracy but increased performance.

[12] 2008 vs. 2006 version used in Cucerzan (2007) We could not get the 2006 version from the author or the Internet.

[13] Since our KB was derived from infoboxes, entities not having an infobox were left out.

References

Javier Artiles, Satoshi Sekine, and Julio Gonzalo. 2008. Web people search: results of the first evaluation and the plan for the second. In *WWW*.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Conference on Computational Linguistics (COLING)*.

Michele Banko and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *Association for Computational Linguistics*.

K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD Management of Data*.

E. Boschee, R. Weischedel, and A. Zamanian. 2005. Automatic information extraction. In *Conference on Intelligence Analysis*.

Razvan C. Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *European Chapter of the Assocation for Computational Linguistics (EACL)*.

Peter Christen. 2006. A comparison of personal name matching: Techniques and practical issues. Technical Report TR-CS-06-02, Australian National University.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Markus Dreyer, Jason Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2009. Scaling Wikipedia-based named entity disambiguation to arbitrary web text. In *WikiAI09 Workshop at IJCAI 2009*.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Knowledge Discovery and Data Mining (KDD)*.

Martin Klein and Michael L. Nelson. 2008. A comparison of techniques for estimating IDF values to generate lexical signatures for the web. In *Workshop on Web Information and Data Management (WIDM)*.

Fangtao Li, Zhicheng Zhang, Fan Bu, Yang Tang, Xiaoyan Zhu, and Minlie Huang. 2009. THU QUANTA at TAC 2009 KBP and RTE track. In *Text Analysis Conference (TAC)*.

Gideon S. Mann and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Conference on Natural Language Learning (CONLL)*.

Andrew McCallum, Kamal Nigam, and Lyle Ungar. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Knowledge Discovery and Data Mining (KDD)*.

Paul McNamee and Hoa Trang Dang. 2009. Overview of the TAC 2009 knowledge base population track. In *Text Analysis Conference (TAC)*.

Paul McNamee, Mark Dredze, Adam Gerber, Nikesh Garera, Tim Finin, James Mayfield, Christine Piatko, Delip Rao, David Yarowsky, and Markus Dreyer. 2009. HLTCOE approaches to knowledge base population at TAC 2009. In *Text Analysis Conference (TAC)*.

Marius Pasca. 2008. Turning web text and search queries into factual knowledge: hierarchical class attribute extraction. In *National Conference on Artificial Intelligence (AAAI)*.

Massimo Poesio, David Day, Ron Artstein, Jason Duncan, Vladimir Eidelman, Claudio Giuliano, Rob Hall, Janet Hitzeman, Alan Jern, Mijail Kabadjov, Stanley Yong, Wai Keong, Gideon Mann, Alessandro Moschitti, Simone Ponzetto, Jason Smith, Josef Steinberger, Michael Strube, Jian Su, Yannick Versley, Xiaofeng Yang, and Michael Wick. 2008. Exploiting lexical and encyclopedic resources for entity disambiguation: Final report. Technical report, JHU CLSP 2007 Summer Workshop.

Gerard Salton and Michael McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Conference on Natural Language Learning (CONLL)*.

Zareen Syed, Tim Finin, and Anupam Joshi. 2008. Wikipedia as an ontology for describing documents. In *Proceedings of the Second International Conference on Weblogs and Social Media*. AAAI Press.

A Discriminative Latent Variable-Based "DE" Classifier for Chinese–English SMT

Jinhua Du and **Andy Way**
CNGL, School of Computing
Dublin City University
{jdu, away}@computing.dcu.ie

Abstract

Syntactic reordering on the source-side is an effective way of handling word order differences. The 的 (DE) construction is a flexible and ubiquitous syntactic structure in Chinese which is a major source of error in translation quality. In this paper, we propose a new classifier model — discriminative latent variable model (DPLVM) — to classify the DE construction to improve the accuracy of the classification and hence the translation quality. We also propose a new feature which can automatically learn the reordering rules to a certain extent. The experimental results show that the MT systems using the data reordered by our proposed model outperform the baseline systems by 6.42% and 3.08% relative points in terms of the BLEU score on PB-SMT and hierarchical phrase-based MT respectively. In addition, we analyse the impact of DE annotation on word alignment and on the SMT phrase table.

1 Introduction

Syntactic structure-based reordering has been shown to be significantly helpful for handling word order issues in phrase-based machine translation (PB-SMT) (Xia and McCord, 2004; Collins et al., 2005; Wang et al., 2007; Li et al., 2007; Elming, 2008; Chang et al., 2009). It is well-known that in MT, it is difficult to translate between Chinese–English because of the different word orders (cf. the different orderings of head nouns and relative clauses). Wang et al. (2007) pointed out that Chinese differs from English in several important respects, such as relative clauses appearing before the noun being modified, prepositional phrases often appearing before the head they modify, etc. Chang et al. (2009) argued that many of the structural differences are related to the ubiquitous Chinese structural particle phrase 的 (DE) construction, used for a wide range of noun modification constructions (both single word and clausal) and other uses. They pointed out that DE is a major source of word order error when a Chinese sentence is translated into English due to the different ways that the DE construction can be translated.

In this paper, we focus on improving the classification accuracy of DE constructions in Chinese as well as investigating its impact on translation quality. From the grammatical perspective, the 的(DE) in Chinese represents the meaning of "noun modification" which generally is shown in the form of a Noun phrase (NP) [A DE B]. A includes all the words in the NP before DE and B contains all the words in the NP after DE. Wang et al. (2007) first introduced a reordering of the DE construction based on a set of rules which were generated manually and achieved significant improvements in translation quality. Chang et al. (2009) extended this work by classifying DE into 5 finer-grained categories using a log-linear classifier with rich features in order to achieve higher accuracy both in reordering and in lexical choice. Their experiments showed that a higher

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 286–294,
Beijing, August 2010

accuracy of the DE classification improved the accuracy of reordering component, and further indirectly improved the translation quality in terms of BLEU (Papineni et al., 2002) scores.

We regard the DE classification as a labeling task, and hence propose a new model to label the DE construction using a discriminative latent variable algorithm (DPLVM) (Morency et al., 2007; Sun and Tsujii, 2009), which uses latent variables to carry additional information that may not be expressed by those original labels and capture more complicated dependencies between DE and its corresponding features. We also propose a new feature defined as "tree-pattern" which can automatically learn the reordering rules rather than using manually generated ones.

The remainder of this paper is organised as follows. In section 2, we introduce the types of word order errors caused by the DE construction. Section 3 describes the closely related work on DE construction. In section 4, we detail our proposed DPLVM algorithm and its adaptation to our task. We also describe the feature templates as well as the proposed new feature used in our model. In section 5, the classification experiments are conducted to compare the proposed classification model with a log-linear model. Section 6 reports comparative experiments conducted on the NIST 2008 data set using two sets of reordered and non-reordered data. Meanwhile, in section 7, an analysis on how the syntactic DE reordering affects word alignment and phrase table is given. Section 8 concludes and gives avenues for future work.

2 The Problem of Chinese DE Construction Translation

Although syntactic reordering is an effective way of significantly improving translation quality, word order is still a major error source between Chinese and English translation. Take examples in Figure 1 as an illustration. The errors of three translation results in Figure 1 are from different MT systems, and many errors relate to incorrect reordering for the 的 (DE) structure.

These three translations are from different Hiero systems. Although Hiero has an inherent reordering capability, none of them correctly reordered "bad reputation" and "middle school" around the DE. Chang et al. (2009) suggested that this is because it is not sufficient to have a formalism which supports phrasal reordering. They claimed it is necessary to have sufficient linguistic modeling, so that the system knows when and how much to rearrange.

Source: 当地(local) 一所(a) 名声不佳(bad reputation) 的(with) 中学(middle school)
Reference: 'a local middle school with a bad reputation'
Team 1: 'a bad reputation of the local secondary school'
Team 2: 'the local a bad reputation secondary school'
Team 3: 'a local stigma secondary schools'

Figure 1: Examples of DE construction translation errors from (Chang et al., 2009)

Figure 2 gives an example illustrating how the reordering of DE construction influences the translation of a Chinese sentence. We can see that if we can properly recognise the DE construction [A DE B] and correctly perform the reordering, we can achieve a closer word order with English and hence a good English translation even it is literal.

Although the Hiero system has a strong reordering capability in its generalised phrases, it still cannot process some complicated and flexible cases of DE construction like those in Figure 1. Therefore, a lot of work has gone into word reordering before decoding so that the Chinese sentences have a closer word order with corresponding English sentences.

3 Related Work on DE Construction

To address the word order problems of the DE construction, Wang et al. (2007) proposed a syntactic reordering approach to deal with structural differences and to reorder source language sentences to be much closer to the order of target language sentences. They presented a set of manually generated syntactic rules to determine whether a 的(DE) construction should be reordered or not before translation, such as "For DNPs consisting of 'XP+DEG', reorder if XP is PP or LCP" etc. (cf. (Wang et al., 2007)). The deficiency of their algorithm is that they did not fully consider the flexibility of the DE construction, as it can be translated in many different ways.

Original:	澳洲	是	[与	北韩	有	邦交]$_A$	的	[少数	国家	之一]$_B$	。
	Aozhou	shi	yu	Beihan	you	bangjiao	DE	shaoshu	guojia	zhiyi	.
	Australia	is	with	North Korea	have	diplomatic relations	that	few	countries	one of	.

Reference: Australia is [one of the few countries] ***that*** [have diplomatic relations with North Korea] .

Reordered: 澳洲 是 [少数 国家 之一]$_B$ 的 [与 北韩 有 邦交]$_A$ 。

Literal Translation: Australia is [one of the few countries] [have diplomatic relations with North Korea] .

Figure 2: An example of DE construction reordering (extended from the original figure in (Chiang, 2005))

Chang et al. (2009) extended the work of (Wang et al., 2007) and characterised the DE structures into 5 finer-grained classes based on their syntactic behaviour. They argued that one possible reason why the 的(DE) construction remains problematic is that previous work has paid insufficient attention to the many ways that the 的 (DE) construction can be translated, as well as the rich structural cues which exist for these translations.

For a Chinese noun phrase [A 的 B], it can be categorized into one of the following five classes (cf. (Chang et al., 2009) for some real examples of each class):

- A B (label: DE$_{AB}$)

  In this category, A on the Chinese side is translated as a pre-modifier of B. In most cases A is an adjectival form.

- B preposition A (label: DE$_{BprepA}$)

  There are several cases that are translated into the form B preposition A.

- A's B (label: DE$_{AsB}$)

  In this class, the English translation is an explicit s-genitive case. This class occurs much less often but is still interesting because of the difference from the of-genitive.

- relative clause (label: DE$_{relc}$)

  In this class, the relative clause would be introduced by a relative pronoun or be a reduced relative clause.

- A preposition B (label: DE$_{AprepB}$)

  This class is another small one. The English translations that fall into this class usually have some number, percentage or level word in the Chinese A.

Chang et al. (2009) used 6 kinds of features for DE classification, namely part-of-speech tag of DE (DEPOS), Chinese syntactic patterns appearing before DE (A-pattern), unigrams and bigrams of POS tags(POS-ngram), suffix unigram and bigram of word (Lexical), Semantic class of words (SemClass) and Re-occurrence of nouns (Topicality). A conditional log-linear classifier (Chang et al., 2009) is trained to classify each DE based on features extracted from the parsed data.

4 Discriminative Probabilistic Latent Variable Model

4.1 Motivation

Based on the discussion so far, we can see that:

- syntactic reordering of the DE construction in Chinese is an effective way to improve the translation quality;
- classifying the DE construction into finer-grained categories could achieve better reordering and translation performance;
- classification accuracy of the DE construction in Chinese has a significant impact on SMT performance.

Driven by these three points, especially the third one, we propose a DPLVM-based classifier to improve classification accuracy. In natural language processing (NLP) such as sequential labeling (Sun and Tsujii, 2009), DPLVM demonstrated excellent capability of learning latent dependencies of the specific problems, and have outperformed several commonly-used conventional models, such as support vector machines, conditional random fields and hidden Markov models.

4.2 DPLVM Algorithm

In this section, we theoretically introduce the definition and mathematical description of the DPLVM algorithm used in NLP tasks (Sun and Tsujii, 2009).

Given a sequence of observations $\mathbf{x} = \{x_1, x_2, \ldots, x_m\}$ and a sequence of labels $\mathbf{y} = \{y_1, y_2, \ldots, y_m\}$, the task is to learn a mapping between $\mathbf{x}$ and $\mathbf{y}$. y_i is a class label and is a member of a set $\mathbf{Y}$ of possible class labels. DPLVM also assumes a sequence of latent variables $\mathbf{h} = \{h_1, h_2, \ldots, h_m\}$, which is hidden in the training examples.

The DPLVM is defined as in (1) (Morency et al., 2007; Sun and Tsujii, 2009):

$$P(\mathbf{y}|\mathbf{x}, \Theta) = \sum_{\mathbf{h}} P(\mathbf{y}|\mathbf{h}, \mathbf{x}, \Theta) P(\mathbf{h}|\mathbf{x}, \Theta) \qquad (1)$$

where Θ are the parameters of the model. It can be seen that the DPLVM equates to a CRF model if it has only one latent variable for each label.

For the sake of efficiency, the model is restricted to have disjoint sets of latent variables associated with each class label. Each h_j is a member in a set $\mathbf{H}_{y_j}$ of possible latent variables for the class label y_j. We define $\mathbf{H}$ as the union of all $\mathbf{H}_{y_j}$ sets, so sequences which have any $h_j \notin \mathbf{H}_{y_j}$ will by definition have $P(\mathbf{y}|\mathbf{x}, \Theta) = 0$, so that the model can be rewritten as in (2):

$$P(\mathbf{y}|\mathbf{x}, \Theta) = \sum_{\mathbf{h} \in \mathbf{H}_{y_1} \times \ldots \mathbf{H}_{y_m}} P(\mathbf{h}|\mathbf{x}, \Theta) \qquad (2)$$

where $P(\mathbf{h}|\mathbf{x}, \Theta)$ is defined by the usual conditional random field formulation, as in (3):

$$P(\mathbf{h}|\mathbf{x}, \Theta) = \frac{\exp \Theta \cdot \mathbf{f}(\mathbf{h}, \mathbf{x})}{\sum_{\forall \mathbf{h}} \exp \Theta \cdot \mathbf{f}(\mathbf{h}, \mathbf{x})} \qquad (3)$$

in which $\mathbf{f}(\mathbf{h}, \mathbf{x})$ is a feature vector. Given a training set consisting of n labeled sequences (x_i, y_i), for $i = 1 \ldots n$, parameter estimation is performed by optimizing the objective function in (4):

$$L(\Theta) = \sum_{i=1}^{n} \log P(y_i|x_i, \Theta) - R(\Theta) \qquad (4)$$

The first term of this equation is the conditional log-likelihood of the training data. The second term is a regularizer that is used for reducing overfitting in parameter estimation.

For decoding in the test stage, given a test sequence $\mathbf{x}$, we want to find the most probable label sequence y^*, as in (5):

$$\mathbf{y}^* = \arg\max_{y} P(\mathbf{y}|\mathbf{x}, \Theta^*) \qquad (5)$$

Sun and Tsujii (2009) argued that for latent conditional models like DPLVMs, the best label path $\mathbf{y}^*$ cannot directly be generated by the Viterbi algorithm because of the incorporation of hidden states. They proposed a latent-dynamic inference (LDI) method based on A^* search and dynamic programming to efficiently decode the optimal label sequence $\mathbf{y}^*$. For more details of the LDI algorithm, refer to (Sun and Tsujii, 2009).

In our experiments, we use the open source toolkit of DPLVM[1] and adapt it to our special requirements based on the different features and scenarios.

4.3 Data and DE Annotation

We use the 5 classes of DE of (Chang et al., 2009) shown in Section 3 to label DE using our DPLVM model. In order to fairly compare the classification performance between that of Chang et al. (2009) and our proposed classifiers, we use the same data sets and conditions to train and test the classifier. The data sets are the Chinese Treebank 6.0 (LDC2007T36) and the English–Chinese Translation Treebank 1.0 (LDC2007T02). For more details about the data sets, refer to (Chang et al., 2009). There are 3523 DEs in the data set, with 543 of them in the “other” category which do not belong to any of the 5 pre-defined classes. In the classification experiments, the “other” class is excluded[2] and 2980 DEs remain, each of which

[1] http://www.ibis.t.u-tokyo.ac.jp/XuSun

[2] In the classification experiments of Chang et al. (2009), the “other” class was excluded, so in order to carry out a

is manually annotated with DE labels for the purpose of classifier training and evaluation.

In order to match the training and testing conditions, we used a parser trained on CTB6 excluding files 1-325 to parse the data sets with DE annotation and extract parse-related features rather than using gold-standard parses (same conditions as in (Chang et al., 2009)). It is worth noting that in the Chinese Treebank, there are two types of POS tag for DE in NPs, namely DEC and DEG. However, as a result of using a trained parser, the POS tags of DE might have other values than DEC and DEG. In our data set, there are four other POS tags, namely {AS, DER, DEV,SP}.

4.4 Labels and Features in DPLVM Model

In our task, we use the 5 class labels of DE constructions in NPs, namely DE_{AB}, DE_{AprepB}, DE_{AsB}, DE_{BprepA}, DE_{relc}.

Note that in the case of the DE construction in Chinese, it is different from traditional sequence labeling tasks such as POS tagging, parsing etc. We only need to label one word in the NP structure, i.e. the 的(DE) in a Chinese NP [A DE B]. Therefore the sequence labeling task becomes efficient and speedy using the DPLVM algorithm.

Based on our task, the mathematical conditions for DE classification in a sequence of [A DE B] are denoted as follows:

- **Sequence of Observations:**

  $\mathbf{x} = x_1, \ldots, x_l, x_{DE}, x_k, \ldots, x_m$, where A=$\{x_1, \ldots, x_l\}$, x_{DE} is the Chinese character 的 (DE), and B=$\{x_k, \ldots, x_m\}$;

- **Set of Labels:**

  $\mathbf{Y} = \{y_i | 1 \le i \le 5\}$, in which the five labels are DE_{AB}, DE_{AprepB}, DE_{AsB}, DE_{BprepA}, DE_{relc}.

- **Latent Variables:**

  $\mathbf{h} = h_1, h_2, \ldots, h_m$, where $m = 3$ in our task.

We employ five features as well in the DPLVM model, namely DEPOS, POS-gram, lexical features, SemClass as well as a new feature: tree-pattern, which is discussed below.

fair comparison, we did so too. For the SMT experiments, however, we kept it.

We did not add the sixth feature used in (Chang et al., 2009) – topicality – in our classifier because we do not consider it to be a very useful in a data set in which the sentences which are randomly stored. In such a corpus, the content between any adjacent sentences are irrelevant in many cases.

The new feature and the templates of all features used in our task are defined as:

DEPOS:
As mentioned in section 4.3, there are 6 kinds of POS tags of DE. Thus, the feature template is defined as in (5):

$$\mathbf{T}_{depos} = \{d_{DE} | d_{DE} \in \mathbf{DP}\}, \text{ where } \mathbf{DP} = \{\text{AS, DEC, DEG,DER,DEV,SP}\}. \quad (5)$$

Tree-pattern:
Chang (2009) used an A-pattern feature which is an indicator function that fires when some syntactic rules are satisfied, such as "A is ADJP if A+DE is a DNP with the form of 'ADJP+DEG'", etc. These rules are induced manually based on the grammatical phenomena at hand. Here we propose a more generalised feature defined as "tree-pattern" to automatically learn the reordering from the training data.

We consider all the sub-tree structures around DE without any word POS tags. For example, consider the parse structure (an example in (Chang et al., 2009)) in (6):

(NP (NP (NR 韩国)) (CP (IP (VP (ADVP (AD 最)) (VP (VA 大)))) (DEC 的)) (NP (NN 投资) (NN 对象国)))))) (6)

where the tree-pattern is "NP NP CP IP VP ADVP VP DEC NP". We do not use the word POS tag (except DE) in this feature, such as NR, AD, VA, etc. The intention of this feature is to enable the classifier to automatically learn the structural rules around DE. Given that the position of DE in the parsing of [A DE B] is i, then the feature template is defined as in (7):

$$\mathbf{T}_{tree\_u} = \{t_{i-l}, \ldots, t_{i-1}, t_i, t_{i+1}, \ldots, t_{i+m}\} \quad (7)$$
$$\mathbf{T}_{tree\_b} = \{t_{i-l}t_{i-l+1}, \ldots, t_{i-1}t_i, t_i t_{i+1}, \ldots, t_{i+m-1}t_{i+m}\}$$

where $\mathbf{T}_{tree\_u}$ is the sequence of unigrams in connection with DE and $\mathbf{T}_{tree\_b}$ is the sequence of bigrams related to DE; l and m are the window sizes of A and B respectively. Generally, we use all the unigrams and bigrams in the parsing of A and B in our experiments. We argue that the important advantage of this feature is that it does not depend on manually generated rules, but instead of learns and generalises the reordering rules from the training data directly.

POS-gram:
The POS-ngram feature adds all unigrams and bigrams in A and B. Given that the position of DE is i in [A DE B], the feature template is defined as in (8):

$$\mathbf{T}_{pos\_u} = \{p_{i-l}, \ldots, p_{i-1}, p_{i+1}, \ldots, p_{i+m}\}$$
$$\mathbf{T}_{pos\_b} = \{p_{i-l}p_{i-l+1}, \ldots, \mathbf{p}_{i-1}\mathbf{p}_{i+1}, \ldots, p_{i+m-1}p_{i+m}\} \quad (8)$$

where $\mathbf{T}_{pos\_u}$ and $\mathbf{T}_{pos\_b}$ are uigrams and bigrams in A and B. In the unigrams, we exclude the POS of DE; in the bigrams, we include a bigram pair across DE.

Some other features such as lexical features, SemClass (cf. (Chang et al., 2009) for details) can be defined using similar feature template.

5 Experiments on DPLVM DE Classifier

In this section, we compare the performance of DE classifiers between the DPLVM and log-linear methods.

The accuracy of classification is defined as in (9):

$$\frac{\text{number of correctly labeled DEs}}{\text{number of all DEs}} \times 100 \quad (9)$$

Phrase Type	Log-linear		DPLVM	
	5-A	2-A	5-A	2-A
DEPOS	54.8	71.0	**56.2**	**72.3**
+A-pattern	67.9	83.7	-	-
+Tree-pattern	-	-	**69.6**	**85.2**
+POS-gram	72.1	84.9	**73.6**	**86.5**
+Lexical	74.9	86.5	**76.4**	**87.9**
+SemClass	75.1	86.7	**76.8**	**88.3**
+Topicality	75.4	86.9	-	-

Table 1: Comparison between the two classifiers on 5-class and 2-class accuracy

Table 1 shows the comparison of accuracy, where "5-A" and "2-A" represent the accuracy of the 5-class and 2-class respectively. The 2-class is the categorised classes of DE in (Wang et al., 2007) which are defined as "reordered" and "non-reordered" categories. It can be seen that our DPLVM classifier outperforms the log-linear classifier by 1.4 absolute (1.86% and 1.61% relative respectively) points both on 5-class and 2-class classifications. Furthermore, we see that the DPLVM achieves significantly better performance than the log-linear model only with the simple feature of "DEPOS". As to the new feature "tree-pattern", we can see that it achieves the improvement of 1.5% compared to the "A-pattern" in terms of the accuracy of "2-A". This improvement attributes to the good learning ability of DPLVM as well as the strong generalisation capability of the tree-pattern feature.

In terms of speed, in our task we only need to label the Chinese character DE in the NP structure [A DE B] rather than label the whole sentence, so that we have a feature matrix of $n \times 1$ for each DE. Accordingly, the DPLVM classifier can run efficiently with low memory usage.

6 Experiments on SMT

6.1 Experimental Setting

For our SMT experiments, we used two systems, namely Moses (Koehn et al., 2007) and Moses-chart. The former is the state-of-the-art PB-SMT system while the latter is a new extended system of the Moses toolkit re-implementing the hierarchical PB-SMT (HPB) model (Chiang, 2005). The alignment is carried out by GIZA++ (Och and Ney, 2003) and then we symmetrized the word alignment using the grow-diag-final heuristic. Parameter tuning is performed using Minimum Error Rate Training (Och, 2003).

The training data contains 2,159,232 sentence pairs.The 5-gram language model is trained on the English part of the parallel training data. The development set (devset) is the NIST MT2006 test set and the test set is the NIST MT2008 "current" test set. All the results are reported in terms of BLEU (Papineni et al., 2002) and METEOR (MTR) (Banerjee and Lavie, 2005) scores.

To run the DE classifiers, we use the Stanford Chinese parser (Levy and Manning, 2003) to parse the Chinese side of the MT training data, the devset and test set.

6.2 Statistics of 5-class DE Annotation

For the DE-annotated MT experiments, after we parse the training data, the devset and the test set, we separately use the two DE classifiers to annotate the DE constructions in NPs in all of the parsed data. Once the DE data are labeled, we pre-process the Chinese data by reordering the sentences only with 的$_{BprepA}$ and 的$_{relc}$ annotations. Table 2 lists the statistics of the DE classes in the MT training data, devset and test set using our DPLVM classifier. "的$_{non}$" denotes the unlabeled 的(DE) which does not belong to any of the 5 classes.

6.3 Experimental Results

The experimental results from the PB-SMT and HPB systems separately using the DPLVM and log-linear classifiers are shown in Table 3.

	PB-SMT			Moses-chart		
	BL	LL	LV	BL	LL	LV
BLEU	22.42	23.47	**23.86**	24.36	24.75	**25.11**
MTR	52.03	53.25	**53.78**	53.37	53.75	**54.21**

Table 3: Experimental results on PB-SMT and Moses-chart. "BL" are the baselines; "LL" indicates the log-linear model-based system; "LV" is our DPLVM method.

The baseline systems indicate that the data is neither categorised into DE classes nor reordered on the Chinese side. We can see that (1) the "LV" method outperformed the "BL" and "LL" by 1.44 absolute (6.42% relative), 0.39 absolute (1.66% relative) BLEU points for PB-SMT, and by 0.75 absolute (3.08% relative), 0.36 absolute (1.45% relative) BLEU points for Moses-chart; (2) the "LV" method achieved the improvements for PB-SMT and Moses-chart in terms of MTR scores compared to the "BL" and "LL" systems. Therefore, using DE classification and reordering on the source-side is helpful in improving translation quality; (3) the results using DPLVM achieve better translation quality than that of the "LL" processed data in terms of BLEU and METEOR (Banerjee and Lavie, 2005) scores, which indirectly shows that DPLVM outperforms the log-linear classification model; and (4) the improvements on both PB-SMT and Moses-chart show that the effectiveness of DE reordering is consistent for different types of MT systems. The results are verified by significance test on 95% confidence interval (Zhang and Vogel, 2004).[3]

7 Analysis

In this section, we plan to evaluate how DE reordering contributes to the improvement of translation quality in two respects, namely word alignment and phrase table.

7.1 Evaluating the Word Alignment

We create a word alignment test set which includes 500 sentences with human alignment annotation, and then add this test set into the MT training corpus. Accordingly, the DE-reordered test set is added into the reordered training corpus as well. Thus, we run GIZA++ using the same configurations for these two sets of data and symmetrize the bidirectional word alignment using grow-diag heuristic. The word alignment of the test set is evaluated with the human annotation using Precision, Recall, F1 and AER measures. The results are reported in Table 4.

	P	R	F1	AER
non-reordered	71.67	62.02	66.49	33.44
reordered	74.02	62.79	67.95	31.98
Gain	**2.35**	**0.77**	**1.46**	**-1.46**

Table 4: Comparison of Precision, Recall, F1 and AER scores of evaluating word alignment on original and reordered data

We can see that in terms of the four measures, the word alignment produced by the reordered data is slightly better than that of the original data. In some sense, we might say that the DE reordering is helpful in improving the word alignment of the training data.

7.2 Evaluating the Phrase Table

Wang et al. (2007) proposed one way to indirectly evaluate the phrase table by giving the same type of input to the baseline and reordered systems,

[3]http://projectile.sv.cmu.edu/research/public/tools/bootStrap/tutorial.htm.

	training		devset		testset	
DE-class	count	percent (%)	count	percent (%)	count	percent (%)
的$_{AB}$	312,679	23.08	523	25.80	453	28.78
的$_{AprepB}$	6,975	0.51	9	0.44	7	0.44
的$_{AsB}$	13,205	0.97	23	1.13	14	0.89
的$_{BprepA}$	658,589	47.31	956	48.05	688	43.71
的$_{relc}$	316,772	23.38	419	20.67	341	21.66
的$_{non}$	46,547	3.44	97	4.79	71	4.51
Total 的	1,354,767	100	2027	100	1574	100

Table 2: The number of different DE classes labeled for training data, devset and testset using the DPLVM classifier

with the consideration that if the reordered system learned a better phrase table, then it may outperform the baseline on non-reordered inputs despite the mismatch and vice versa. However, they did not settle the question as to whether the reordered system can learn better phrase tables.

We also try to use the idea of Wang et al (2007) to carry out the phrase table evaluation on PB-SMT,[4] i.e. we tune the baseline on a reordered devset and then evaluate on a reordered test set; tune the reordered system on a non-reordered devset and then evaluate on a non-reordered test set. The results are shown in Table 5.

Testset	baseline	reordered	
		LL	DPLVM
non-reordered set	22.42	22.76	22.85
reordered set	23.36	23.47	23.86

Table 5: Comparison of BLEU scores in matched and mismatched conditions on PB-SMT.

We find that (1) given the non-reordered test set, the DE reordered system performs better than the baseline system, which is consistent when different DE classifiers are applied; (2) given the reordered test set system, the reordered set produces a better result than the baseline, which is also consistent when different DE classifiers are applied; and (3) the results from the DPLVM-based reordered data are better than those from the LL-based reordered data. From the comparison, one might say that the reordered system was learned a better phrase table and the reordered test set addresses the problem of word order.

To sum up, from the SMT results and the evaluation results on the word alignment and the phrase table, we can conclude that the DE reordering methods contribute significantly to the improvements in translation quality, and it also implies that using DE reordered data can achieve better word alignment and phrase tables.

8 Conclusions and Future Work

In this paper, we presented a new classifier: a DPLVM model to classify the Chinese 的(DE) constructions in NPs into 5 classes. We also proposed a new and effective feature – tree-pattern – to automatically learn the reordering rules using the DPLVM algorithm. The experimental results showed that our DPLVM classifier outperformed the log-linear model in terms of both the classification accuracy and MT translation quality. In addition, the evaluation of the experimental results in section 7 indicates that the DE-reordering approach is helpful in improving the accuracy of the word alignment, and can also produce better phrase pairs and thus generate better translations.

As for future work, firstly we plan to examine and classify the DE constructions in other syntactic structures such as VP, LCP etc. Secondly, we plan to apply the DE-annotated approach in a syntax-based MT system (Zollmann and Venugopal, 2006) and examine the effects. We also intend to improve the classification accuracy of the DE classifier with richer features to further improve translation quality.

[4]The phrases in HPB systems are different from those in PB-SMT because they are variable-based, so we evaluate the hierarchical phrases in (Du and Way, 2010)

Acknowledgment

Many thanks to Dr. Pi-Chuan Chang for providing the source code of her DE classifier and manually DE-annotated training data as well as valuable instruction in their use. Thanks also to Dr. Xu Sun for the source code of his Latent Variable classifier together with help in their use. This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, ACL-2005*, pages 65–72.

Pi-Chuan Chang, Dan Jurafsky and Christopher D. Manning. 2009 Disambiguating "DE" for Chinese-English machine translation. In *Proceedings of the Fourth Workshop on SMT*, pages 215–223.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL'05*, pages 263–270.

Michael Collins, Philipp Koehn, and Ivona Kucerova. newblock 2005. Clause restructuring for statistical machine translation. In *Proceedings of ACL05*, pages 531–540.

Jinhua Du and Andy Way. 2010. The impact of source-side syntactic reordering on hierarchical phrase-based SMT. In *Proceedings of the 14th Annual conference of the European Association for Machine Translation*, Saint-Raphael, France.

Jakob Elming. 2008. Syntactic reordering integrated with phrase-based SMT. In *Proceedings of ACL-08 SSST-2*, pages 46–54.

Philipp Koehn, Hieu Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, Wade Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *demonstration session of ACL'07*, pages 177–180.

Roger Levy and Christopher Manning. 2003. Is it harder to parse Chinese, or the Chinese treebank? In *Proceedings of ACL'03*, pages 439–446.

Chi-Ho Li, Dongdong Zhang, Mu Li, Ming Zhou, Minghui Li and Yi Guan. 2007. A probabilistic approach to syntax-based reordering for statistical machine translation. In *proceedings of the ACL'07*, pages 720–727.

Louis-Philippe Morency, Ariadna Quattoni and Trevor Darrell. 2007. Latent-dynamic Discriminative Models for Continuous Gesture Recognition. In *proceedings of CVPR'07*, pages 1–8.

Franz Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of ACL'03*, pages 160–167.

Franz Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the ACL-02*, pages 311–318.

Xu Sun and Jun'ichi Tsujii. 2009. Sequential Labeling with Latent Variables: An Exact Inference Algorithm and An Efficient Approximation. In *Proceedings of The European Chapter of the Association for Computational Linguistics (EACL'09)*, pages 772-780.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of EMNLP-CoNLL*, pages 737–745.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of Coling 2004*, pages 508–514.

Ying Zhang and Stephan Vogel. 2004. Measuring Confidence Intervals for the Machine Translation Evaluation Metrics. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI)*, pages 85–94.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings of HLT-NAACL 2006: Proceedings of the Workshop on Statistical Machine Translation*, New York, pages 138–141.

An Empirical Study on Learning to Rank of Tweets

[1]Yajuan Duan* [2]Long Jiang [2]Tao Qin [2]Ming Zhou [2]Heung-Yeung Shum
[1]Department of Computer Science and Technology
University of Science and Technology of China
[2]Microsoft Research Asia
{v-yaduan,longj,taoqin,mingzhou,hshum}@microsoft.com

Abstract

Twitter, as one of the most popular micro-blogging services, provides large quantities of fresh information including real-time news, comments, conversation, pointless babble and advertisements. Twitter presents tweets in chronological order. Recently, Twitter introduced a new ranking strategy that considers popularity of tweets in terms of number of retweets. This ranking method, however, has not taken into account content relevance or the twitter account. Therefore a large amount of pointless tweets inevitably flood the relevant tweets. This paper proposes a new ranking strategy which uses not only the content relevance of a tweet, but also the account authority and tweet-specific features such as whether a URL link is included in the tweet. We employ *learning to rank* algorithms to determine the best set of features with a series of experiments. It is demonstrated that whether a tweet contains URL or not, length of tweet and account authority are the best conjunction.

1 Introduction

Twitter provides a platform to allow users to post text messages known as tweets to update their followers with their findings, thinking and comments on some topics (Java et al., 2007). The searched tweets are presented by Twitter in chronological order except the first three, which are ranked by considering popularity of tweets in terms of the number of retweets.

This ranking method, however, has not taken into account the content relevance and twitter account; inevitably, a large amount of pointless tweets (Pear Analytics, 2009) may flood the relevant tweets. Although this ranking method can provide fresh information to tweet users, users frequently expect to search relevant tweets to the search queries. For example, consider someone researching consumer responses toward the iPad. He or she would like to find tweets with appropriate comments such as *iPad is great* or *you can find many useful features of iPad,* rather than tweets with irrelevant comment, even if they are most recent or popular.

Moreover, neither Twitter's current chronological order based ranking nor the recently introduced popularity based ranking can avoid spam. A developer can accumulate hundreds of thousands of followers in a day or so. At the same time, it is not difficult for spammers to create large quantities of retweets. By contrast, content relevance ranking can effectively prevent spammers from cheating. Different from ranking tweets through chronological order and popularity, a content relevance strategy considers many characteristics of a tweet to determine its ranking level. Thus it is difficult for spammers to break the ranking system by simple methods such as increasing retweet count or number of followers.

In this paper, we propose a method to rank the tweets which outputs the matched tweets based on their content relevance to the query. We

* The work was done when the first author was intern at Microsoft Research Asia

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 295–303,
Beijing, August 2010

investigate the effects of content features and non-content features and produce a ranking system by a *learning to rank* approach.

With a series of experiments, we determined the best set of features and analyzed the effects of each of individual feature. We provide empirical evidence supporting the following claims,

- Account authority, length of tweet and whether a tweet contains a URL are the top three effective features for tweet ranking, where containing a URL is the most effective feature.
- We find an effective representation of account authority: the number of times the author was listed by other users. We find through experiments that this representation is better than the widely adopted number of followers.

2 Related Work

2.1 Real-time Search

At present, a number of web sites offer the so-called real-time search service which mainly returns real-time posts or shared links, videos and images obtained from micro-blogging systems or other medium according to the user's query. We investigate the ranking method used by these web sites. From their self-introduction page, we find four main criteria for ranking real-time posts. They are posting time, account authority, topic popularity and content relevance.

Specifically, Twitter maintains a specialized search engine which ranks tweets according to posting time and topic popularity. In addition, Google, Twazzup[2] and Chirrps[3] rank real-time tweets by posting time. While the last one also ranks tweets by popularity, which is measured by retweet count.

Tweefind[4] ranks search result according to authority of authors which depends on how popular, relevant, and active the author is. Additionally, Twitority[5] rank tweets by author authority as well.

Bing and CrowdEye[6] rank tweets by posting time or content relevance. Bing takes authors authority, retweet count and freshness into consideration while measuring the relevance. To determine the relevance of a tweet, CrowdEye considers a number of factors including content relevance and author influence which appears to rely heavily on the number of followers an author has. It turns out that the number of followers is not a very reasonable measure of the influence of an account according to our experimental results.

2.2 Twitter Recommendation

Besides tweet search, recently some researchers have focused on twitter recommendation system.

Chen et al. (2010) presented an approach to recommend URLs on Twitter as a means to better direct user attention in information streams. They designed the recommender taking three separate dimensions into consideration: content source, topic interest and social voting.

Sun et al. (2009) proposed a diffusion-based micro-blogging recommendation framework aiming to recommend micro-blogs during critical events via optimizing story coverage, reading effort and delay time of a story. The key point of this method is to construct an exact diffusion graph for micro-blogging, which is difficult due to the presence of extensive irrelevant personal messages and spam.

2.3 Blog Search and Forum Search

Another related topic is blog search and forum search. Recently, many approaches for blog search and forum search have been developed, which include *learning to rank* methods and link-based method.

Learning to rank approach

Xi et al. (2004) used features from the thread trees of forums, authors, and lexical distribution within a message thread and then applied Linear Regression and Support Vector Machine (SVM) to train the ranking function. Fujimura et al. (2005) exploited provisioning link and evaluation link between bloggers and blog entries, and scored each blog entry by weighting the hub and authority scores of the bloggers.

Link-Based approach

---

2 Twazzup: http://www.twazzup.com/
3 Chirrps: http://chirrps.com/
4 Tweefind: http://www.tweefind.com/
5 Twitority: http://www.twitority.com/
6 CrowdEye: http://www.crowdeye.com/

Kritikopoulos et al. (2006) introduced similarities among bloggers and blogs into blog ranking. This method enabled the assignment of a higher score to the blog entry published by a blogger who has already accepted a lot of attention. Xu and Ma (2006) built a topic hierarchy structure through content similarity. Liu et al. (2007) presented a newsgroup structure-based approach PostRank which built posting trees according to response relationship between postings.

Chen et al. (2008) proposed a posting rank algorithm which built link graphs according to co-replier relationships. This kind of method exploits different types of structures among postings and improved the performance of traditional link-based ranking algorithm for forum search. However, it is difficult to rank postings which only have a few words simply based on content by using FGRank algorithm. And PostingRank approach relies too much on reply relations which are more likely to suffer from topic excursion.

Although approaches proposed above perform effectively in forum search and blog search, they are not appropriate for twitter search because tweets are usually shorter and more informal than blogs. Furthermore, it does not have the explicit hierarchy structure of newsgroup messages on forums. In addition, tweets possess many particular characteristics that blog and forum do not have.

3 Overview of Our Approach

To generate a good ranking function which provides relevant search results and prevents spammers' cheating activities, we analyze both content features and authority features of tweets and determine effective features. We adopt *learning to rank* algorithms which have demonstrated excellent power in addressing various ranking problems of search engines.

3.1 Learning to Rank Framework

Learning to Rank is a data-driven approach which integrates a bag of features in the model effectively. Figure 1 shows the paradigm of learning for tweet ranking.

At the first step, we prepare the training and test corpus as described in Section 5. Then we extract features from the training corpus. RankSVM algorithm (Joachims Thorsten, 1999) is used to train a ranking model from the training corpus. Finally, the model is evaluated by the test corpus.

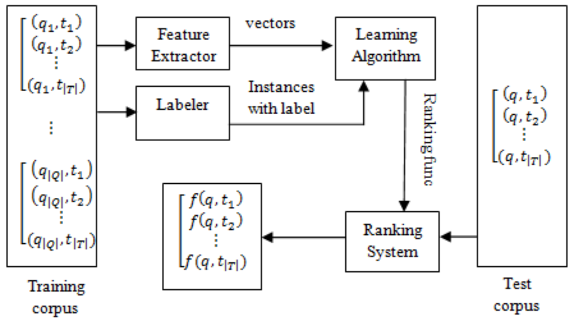


Figure 1. General Paradigm of Learning for Tweets Ranking

3.2 Features for Tweets Ranking

One of the most important tasks of a *learning to rank* system is the selection of a feature set. We exploit three types of features for tweet ranking.

1) *Content relevance features* refer to those features which describe the content relevance between queries and tweets.
2) *Twitter specific features* refer to those features which represent the particular characteristics of tweets, such as retweet count and URLs shared in tweet.
3) *Account authority features* refer to those features which represent the influence of authors of the tweets in Twitter (Leavitt et al., 2009).

In the next section, we will describe these three types of features in detail.

4 Feature Description

4.1 Content Relevance Features

We used three content relevance features, Okapi BM25 (Robertson et al., 1998), similarity of contents and length of tweet.

Okapi BM25 score measures the content relevance between query Q and tweet T. The standard BM25 weighting function is:

$$\mathrm{BM25}(T,Q)=\sum_{q_i\in Q}\frac{\mathrm{IDF}(q_i)\cdot \mathrm{tf}(q_i,T)\cdot(k_1+1)}{\mathrm{tf}(q_i,T)+k_1\left(1-b+b\frac{\mathrm{Length}(T)}{\mathrm{avg}_{\mathrm{length}}}\right)} \quad (1)$$

where Length(T) denotes the length of T and avg_{length} represents average length of tweet in corpus. IDF(q_i) is Inverse Document Frequency.

Similarity of contents estimates the popularity of documents in the corpus (Song et al., 2008). In our case, it measures how many tweets of the query are similar in content with the current tweet. We calculate a cosine similarity score for every pair of tweets, and the final similarity score for tweet T_i in T_{Q_k} is computed by the following formula:

$$\mathrm{Similarity}(T_i) = \frac{1}{|T_{Q_k}|-1} \sum_{T_j \in T_{Q_k}, j \neq i} \frac{TV_i \cdot TV_j}{|TV_i| \cdot |TV_j|} \quad (2)$$

Where TV_i represents the TFIDF vector of T_i and T_{Q_k} refers to tweets collection of query Q_k.

Length is measured by the number of words that a tweet contains. Intuitively, a long sentence is apt to contain more information than a short one. We use length of tweet as a measure of the information richness of a tweet.

4.2 Twitter's Specific Features

Tweets have many special characteristics. We exploit these characteristics and extract six twitter specific features as listed in Table 1.

Feature	Description
URL	Whether the tweet contains a URL
URL Count	Frequency of URLs in corpus
Retweet Count	How many times has this tweet been retweeted
Hash tag Score	Sum of frequencies of the top-n hash tags appeared in the tweet
Reply	Is the current tweet a reply tweet
OOV	Ratio of words out of vocabulary

Table 1. Twitter Specific Features

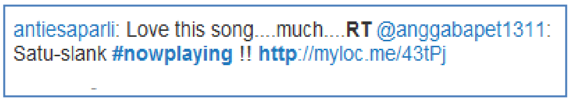


Figure 2. A Tweet Example

URL & URL Count: Twitter allows users to include URL as a supplement in their tweets. The tweet in Figure 2 contains URL http://myloc.me/43tPj which leads to a map indicating where the publisher located.

URL is a binary feature. It is assigned 1 when a tweet contains at least one URL, otherwise 0. URL Count estimates the number of times that the URL appears in the tweet corpus.

Retweet Count: Twitter users can forward a tweet to his or her followers with or without modification on the forwarded tweets, which is called retweet on Twitter. A retweeted tweet usually includes an *RT* tag. Generally, sentences before *RT* are comments of the retweeter and sentences after *RT* are the original content, perhaps with some modifications. Here we only consider tweets including *RT* with the original content unmodified. Retweet count is defined as the number of times a tweet is retweeted. In Figure 2, original tweet *Satu-slank #nowplaying !! http://myloc.me/43tPj* is retweeted once.

Hash Tag Score: Publishers are allowed to insert hash tags into their tweets to indicate the topic. In Figure 2, *#nowplaying* is a hash tag. We collect hash tags appearing in the tweets of every query and sort them in descending order according to frequency. Tag frequency for tweet T_i of query Q_k is computed from normalized frequency of top-n tags.

$$\mathrm{TagScore}(T_i) = \frac{1}{z_k} \sum_{\substack{j=1, tag_j \in Tag_{Q_k} \\ tag_j \in T_i}}^{n} \mathrm{freq}(tag_j) \quad (3)$$

Where z_k is the normalization factor. $\mathrm{freq}(tag_j)$ represents the frequent of tag_j in corpus. And Tag_{Q_k} denotes the tag collection extracted from T_{Q_k}.

Reply: This is a binary feature. It is 1 when the tweet is a reply and 0 otherwise. A tweet starting with a twitter account is regarded as a reply tweet in our experiment. Figure 3 shows an example.

Figure 3. Reply Tweet

OOV: This feature is used to roughly approximate the language quality of tweets. Words out of vocabulary in Twitter include spelling errors and named entities. According to a small-scale investigation, spelling errors account for more than 90% of OOVs excluding capitalized words, tags, mentions of users and

URLs. We use a dictionary with 0.5 million entries to compute the ratio of OOVs in a tweet.

$$\text{Quality(T)} = \frac{\text{\# of OOVs in T}}{\text{Length(T)}} \qquad (4)$$

4.3 Account Authority Features

There are three important relations between users in Twitter: follow, retweet, and mention. Additionally, users are allowed to classify their followings into several lists based on topics. We measured the influence of users' authorities on tweets based on the following assumptions:

- Users who have more followers and have been mentioned in more tweets, listed in more lists and retweeted by more important users are thought to be more authoritative.
- A tweet is more likely to be an informative tweet rather than pointless babble if it is posted or retweeted by authoritative users.

PageRank algorithm for calculating popularity score for users.
Input: Directed Graph G of retweet relationship
Damping factor e.
Output: popularity score for each user
Procedure:
Step 1: popularity score of all users are initialized as $1 - e$.
Step 2: update the popularity score for users.

$$PScore_{t+1}(v_i) = 1 - e + e \cdot \sum_{v_j \in R_{v_i}} \frac{PScore_t(v_j)RN_{ij}}{N_j}$$

R_{v_i} denotes the collection of users who retweeted v_i's tweet.
RN_{ij} is the number of times v_i has been retweeted by v_j.
N_j is the number of users whose tweets v_j has retweeted.
Step 3: Repeat the second step until all popularity scores will never change.

Figure 4. PageRank Algorithm for Calculating Popularity Score for Users

In order to distinguish the effect of the three relations, we computed four scores for each user representing the authority independently.

- Follower Score: number of followers a user has.
- Mention Score: number of times a user is referred to in tweets.
- List Score: number of lists a user appears in.
- Popularity Score: computed by PageRank algorithm (Page et al., 1999) based on retweet relations.

Following the retweet relationship among users, we construct a directed graph G (V, E). In our experiments, G is built from a tweet collection including about 1.1 million tweets. V denotes twitter users that appear in training examples. E is a set of directed edges. If author v_i published the tweet t_k, and author v_j retweeted t_k after v_i, there exists an edge from v_j to v_i. We call v_i original author and v_j retweeter. Figure 4 shows the PageRank algorithm for calculating popularity scores for twitter users. In our experiment, damping factor e was set to 0.8. Like Dong et al. (2010) did, we define three subtypes for each account authority score. Table 2 presents features of account authority we use.

Feature	Description
Sum_follower	Sum of follower scores of users who published or retweeted the tweet
Sum_popularity	Sum of popularity scores of users who published or retweeted the tweet
Sum_mention	Sum of mention scores of users who published or retweeted the tweet
Sum_list	Sum of list scores of users who published or retweeted the tweet
First_follower	Follower score of the user who published the tweet
First_popularity	Popularity score of the user who published the tweet
First_mention	Mention score of the user who published the tweet
First_list	List score of the user who published the tweet
Important_follower	The highest follower score of the user who published or retweeted the tweet
Important_popularity	The highest popularity score of the user who published or retweeted the tweet
Important_mention	The highest mention score of the user who published or retweeted the tweet
Important_list	The highest list score of the user who published or retweeted the tweet

Table 2. Account Authority Features for tweet

5 Experiment Data and Evaluation

We introduce the data we used in experiment and the evaluation metrics in this section.

5.1 Data

We analyze 140 hot searches on CrowdEye within a week. They consist of big events, famous persons, new products, festivals, movies and so on. The most frequent types of hot searches, which account for more than 81% of all hot searches, are as follows:

- News: news about public figures and news related to some places.
- Products: character description, promotion information and comments about products.
- Entertainment: mainly about movies, including film reviews and introductions about plots.

We select 20 query terms as shown in Table 3, including 5 persons, 5 locations, 5 products and 5 movie names. Specifically, Locations are sampled from a list of American cities. Person names come from the hot search and hot trends provided by Twitter and CrowdEye. Products are sampled from the popular searches of 35 product categories on eBay. And movies are selected from a collection of recommended movies from 2005 to 2010. We crawl 162,626 English tweets for the selected queries between March 25, 2010 and April 2, 2010 from Twitter Search. After removing the repeated ones, 159,298 tweets remained.

Query type	**Query terms**
Locations	New York, Nashville, Denver, Raleigh, Lufkin
Person Names	Obama, Bill Clinton, James Cameron, Sandra Bullock, LeBron James
products	Corvette, iPad, Barbie, Harry Potter, Windows 7
Movies	The Dark Knight, up in the air, the hurt locker, Batman Begins, Wall E

Table 3. 20 Query Terms

Retweets are forwardings of corresponding original tweets, sometimes with comments of retweeters. They are supposed to contain no more information than the original tweets, therefore they drops out of ranking in this paper.

We sample 500 tweets for each query from its original tweets collection and ask a human editor to label them with a relevance grade. In order to ensure the annotation is reasonable, we set multiple search intentions for each query referring to the topics arising in the tweets about the query in the corpus. Specifically, for Locations, tweets describing news related to the location are relevant. For people, what they have done and the comments about them are regarded as relevant information. For products, tweets including feature description, promotion and comments are considered relevant. And for movies, tweets about comment on the movies, show time and tickets information are preferred. We apply four judgment grades on query-tweet pairs: excellent, good, fair and bad. According to the statistics, about half of the tweets in the experiment data are labeled as bad. Table 4 presents the distribution for all grades.

Grade	Excellent	Good	Fair	Bad
Percentage	20.9%	10.9%	16.9%	51.3%
Min	2.4%	1.8%	4.0%	8.0%
Max	69.8%	23.2%	54.4%	81.0%

Table 4. Tweet Distribution of Each Grade

5.2 Evaluation Metrics

There are several metrics that are often used to measure the quality of rankings. In this paper, we use Normalized Discount Cumulative Gain (NDCG) which can handle multiple levels of relevance as the evaluation metrics (Jarvelin and Kekalainen, 2002).

6 Results

Five-fold cross-validation was used in our experiments. We choose tweets of sixteen queries (four from each query type) as the training data. The remaining tweets are divided into evaluation data and validation data equally.

6.1 Learning to Rank for Tweet Ranking

We learn a ranking model by using a RankSVM algorithm based on all features we extracted, which is denoted as RankSVM_Full. In the experiment, a toolkit named svm^{struct} [7] implemented by Thorsten Joachims is used. Figure 5 shows the comparison between our method which integrates three types of features and ranking through chronological order, account authority, and content relevance individually.

In this experiment, Content Relevance is measured by BM25 score. And Account

[7] SVM^{struct}: http://svmlight.joachims.org/svm_struct.html

Authority is approximated by the number of followers of the user. Figure 5 illustrates that ranking through content relevance is not as effective as other methods. This is because our work is essentially re-ranking on the result of Twitter Search. Hence almost all tweets include the query term which makes it difficult to distinguish them by BM25 score. Figure 5 also reveals that account authority is useful for ranking tweet relevance; it outperforms ranking through chronological order and is competitive to our model trained from all features. This agrees with the assumption we made about the influence of user authorities on tweets.

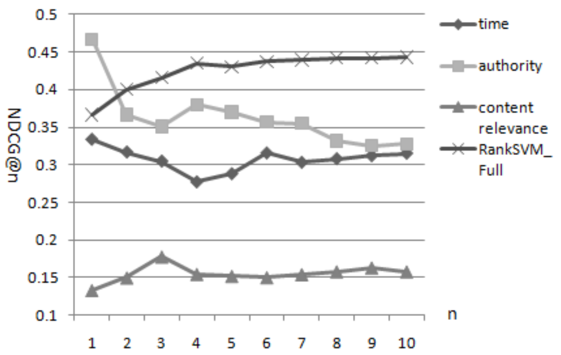


Figure 5. Performance of Four Ranking Methods

6.2 Feature Selection

As the RankSVM_Full underperforms against some models trained from subsets of features, we use an advanced greedy feature selection method and find the best feature conjunction to improve the performance of RankSVM_full. Figure 6 shows the feature selection approach.

Although greedy feature selection approach is commonly used in many problems, it does not work efficiently in addressing this problem partly for data sparseness. It is always blocked by a local optimum feature set. In order to resolve this problem, we first generate several feature sets randomly and run the greedy selection algorithm based the best one among them. Finally, we find the best feature conjunction composed by *URL, Sum_mention, First_List, Length,* and *Important_follower*, from which a model is learnt denoted as RankSVM_Best. Figure 7 illustrates that this model outperforms RankSVM_Full by about 15.3% on NDCG@10.

An advanced greedy feature selection algorithm.
Input: All features we extracted.
Output: the best feature conjunction *BFC*
Procedure:
Step1: Randomly generate 80 feature set *F*.
Step 2: Evaluate every feature set in *F* and select the best one denoted by *RBF*.
Features excluded those in *RBF* are denoted as *EX_RBF*
Step 3: t = 0,*BFC*(t)=*RBF*;
Repeat
Foreach feature in *EX_RBF*
If Evaluation(*BFC*) < Evaluation(*BFC*, feature)
BFC(t+1) = {*BFC*(t), feature}
EX_RBF(t+1) = *EX_RBF*(t) – {feature}
While *BFC*(t+1) ≠ *BFC*(t)
Note: Evaluation(*BFC*) refers to the performance of ranking function trained from features in *BFC* on validation data.

Figure 6. Advanced Greedy Feature Selection Algorithm

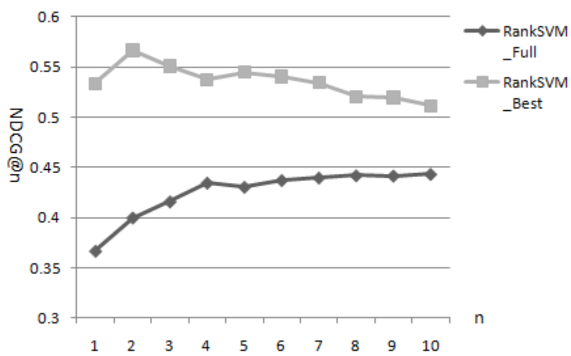


Figure 7. Comparison between RankSVM_Full and RankSVM_Best

We conduct a paired t-test between RankSVM_Best and each of other four ranking methods on NDCG@10 of ten test queries. The results demonstrate that RankSVM_Best outperforms ranking through time, account authority and content relevance respectively with a significance level of 0.01, and RankSVM_Full with a level of 0.05.

6.3 Feature Analysis

We are interested in which features in particular are highly valued by our model for tweet ranking. We evaluate the importance of each feature by the decrement of performance when removing the feature measured from RankSVM_Best. Figure 8 reveals the importance of each feature in our model.

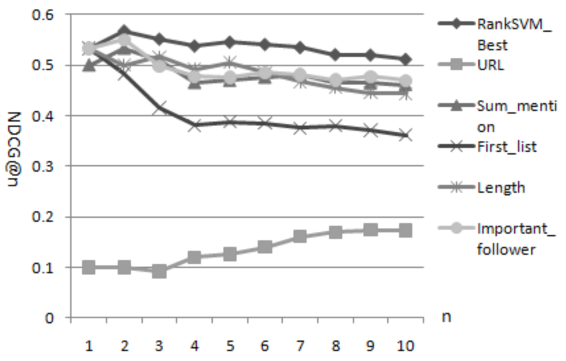


Figure 8. Importance of Each Feature

We observe from Figure 8 that URL is very important for our model; without it the performance declines seriously (with a significance level of 0.001). The reason may be that URLs shared in tweets, which provide more detailed information beyond the tweet's 140 characters, may be relevant to the query at a high probability.

Another useful feature is the number of lists that the author of the tweet has been listed in. The performance of ranking decreases with a significance level of 0.05 when removing it from the best feature combination. However, other features do not show significant contribution.

7 Discussion

Our experiment in section 6.2 demonstrates that features such as Hash tag Score and Retweet Count are not as effective as expected. This may be due to the small size of training data. We present an approach to learn an effective tweets ranker in a small dataset through feature selection. However, 20 queries are not sufficient to train a powerful ranker for Twitter.

In this study, to minimize the annotation effort, for each test query, we only annotate the tweets containing the query (returned by Twitter Search) and then used them for evaluation. With this kind of evaluation, it is hard to completely evaluate the significance of some features, such as content relevance features. In the future, we will select more queries including both hot searches and long tail searches, and select tweets for annotation directly from the twitter firehose.

There is also an opportunity for more accurate retweet relation detection in our work. At present, we just identify the retweet whose original tweet has not been modified, which leaves out a fair amount of retweet information. We would need to develop a more precise retweet relation detection method.

8 Conclusion

In this paper, we study three types of tweet features and propose a tweet ranking strategy by applying learning to rank algorithm. We find a set of most effective features for tweet ranking. The results of experiments demonstrate that the system using *Sum_mention*, *First_list*, *Important_follower*, *length* and *URL* performs best. In particular, whether a tweet contains a URL is the most effective feature. Additionally, we find in the experiments that the number of times the account is listed by other users is an effective representation of account authority and performs better than the number of followers that is widely used in previous work.

There are many aspects we would like to explore in the future. First, this research is based on the search results returned from Twitter which contains the input query. The tweets not containing the queries are not returned. We will explore query expansion approaches to improve the recall of the search results. We did not consider spam issues in the ranking process. However, spam filtering is important to all types of search engines. We will explore the impacts of spam and work out a spam filtering approach.

References

Chen Jilin, Rowan Nairn, Les Nelson, Michael Bernstein, and Ed H. Chi. 2010. Short and Tweet: Experiments on Recommending Content from Information Streams. *In the Proceedings of the 28th International conference on Human Factors in Computing Systems*, Pages: 1185-1194.

Chen Zhi, Li Zhang, Weihua Wang. 2008. PostingRank: Bringing Order to Web Forum Postings. *In the proceedings of the 4th Asia Information Retrieval Symposium*, Pages: 377-384.

Dong Anlei, Ruiqiang Zhang, Pranam Kolari, Jing Bai, Fernando Diaz, Yi Chang, Zhaohui Zheng, and Hongyuan Zha. 2010. Time of the essence: improving recency ranking using Twitter data. *In the proceedings of the 19th International Conference on World Wide Web*, Pages: 331-340.

Fujimura Ko, Takafumi Inoue, and Masayuki Sugisaki. 2005. The EigenRumor Algorithm for Ranking Blogs. *In the proceedings of the 2nd Annual Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics, World Wide Web*.

Jarvelin Kalervo, and Jaana Kekalainen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, Volume 20, Pages: 422-446.

Java Akshay, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. Why we twitter: Understanding Microblogging Usage and Communities. *In the proceedings of the 9th International Workshop on Knowledge Discovery on the Web and the 1st International Workshop on Social Networks Analysis*. Pages: 118-138.

Joachims Thorsten. 1999. Making Large-Scale SVM Learning Practical. *Advances in Kernel Methods: Support Vector Learning*, Pages: 169-184.

Pear Analytics. 2009. Twitter Study-August 2009.

Kritikopoulos Apostolos, Martha Sideri, and Iraklis Varlamis. 2006. BlogRank: Ranking Weblogs Based on Connectivity and Similarity Features. *In the proceedings of the 2nd International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*.

Leavitt Alex, Evan Burchard, David Fisher, and Sam Gilbert. 2009. The Influentials: New Approaches for Analyzing Influence on Twitter. *A publication of the Web Ecology Project*.

Liu Hongbo, Jiahai Yang, Jiaxin Wang, Yu Zhang. 2007. A Link-Based Rank of Postings in Newsgroup. *In the proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition*, Pages: 392-403.

Page Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking: Bring Order to the Web. *Technical report*, Stanford University.

Robertson Stephen E., Steve Walker, and Micheline Hancock-Beaulieu. 1998. Okapi at TREC-7: Automatic Ad Hoc, Filtering, VLC and Interactive. *In the Proceedings of the 7th Text Retrieval Conference*. Pages: 199-210

Song Young-In, Chin-Yew Lin, Yunbo Cao, and Hae-Chang Rim. 2008. Question Utility: A Novel Static Ranking of Question Search. *In the Proceedings of the 23rd AAAI Conference on Artificial Intelligence*. Pages: 1231-1236

Sun Aaron R., Jiesi Cheng, and Daniel D. Zeng. 2009. A Novel Recommendation Framework for Micro-blogging based on Information Diffusion. *In the proceedings of the 19th Workshop on Information Technologies and Systems*.

Xi Wensi, Jesper Lind, and Eric Brill. 2004. Learning effective ranking functions for newsgroup search. *In the proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Pages: 394-401

Xu Gu, and Ma Wei-Ying. 2006. Building Implicit Links from Content for Forum Search. *In the proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Pages: 300-307.

Translation Model Generalization using Probability Averaging for Machine Translation

Nan Duan[1]**, Hong Sun**
School of Computer Science and Technology
Tianjin University
v-naduan@microsoft.com
v-hongsun@microsoft.com

Ming Zhou
Microsoft Research Asia
mingzhou@microsoft.com

Abstract

Previous methods on improving translation quality by employing multiple SMT models usually carry out as a second-pass decision procedure on hypotheses from multiple systems using extra features instead of using features in existing models in more depth. In this paper, we propose *translation model generalization* (TMG), an approach that updates probability feature values for the translation model being used based on the model itself and a set of auxiliary models, aiming to enhance translation quality in the first-pass decoding. We validate our approach on translation models based on auxiliary models built by two different ways. We also introduce novel probability variance features into the log-linear models for further improvements. We conclude that our approach can be developed independently and integrated into current SMT pipeline directly. We demonstrate BLEU improvements on the NIST Chinese-to-English MT tasks for single-system decodings, a system combination approach and a model combination approach.

1 Introduction

Current research on Statistical Machine Translation (SMT) has made rapid progress in recent decades. Although differed on paradigms, such as phrase-based (Koehn, 2004; Och and Ney, 2004), hierarchical phrase-based (Chiang, 2007) and syntax-based (Galley *et al.*, 2006; Shen *et al.*, 2008; Huang, 2008), most SMT systems follow the similar pipeline and share common translation probability features which constitute the principal components of translation models. However, due to different model structures or data distributions, these features are usually assigned with different values in different translation models and result in translation outputs with individual advantages and shortcomings.

In order to obtain further improvements, many approaches have been explored over multiple systems: system combination based on confusion network (Matusov *et al.*, 2006; Rosti *et al.*, 2007; Li *et al.*, 2009a) develop on multiple *N*-best outputs and outperform primary SMT systems; consensus-based methods (Li *et al.*, 2009b; DeNero *et al.*, 2010), on the other hand, avoid the alignment problem between translations candidates and utilize *n*-gram consensus, aiming to optimize special decoding objectives for hypothesis selection. All these approaches act as the second-pass decision procedure on hypotheses from multiple systems by using extra features. They begin to work only after the generation of translation hypotheses has been finished.

In this paper, we propose *translation model generalization* (TMG), an approach that takes effect during the first-pass decoding procedure by updating translation probability features for the translation model being used based on the model itself and a set of auxiliary models. Bayesian Model Averaging is used to integrate values of identical features between models. Our contributions mainly include the following 3 aspects:

- *Alleviate the model bias problem based on translation models with different paradigms.* Because of various model constraints, translation models based on different paradigms could have individual biases. For instance, phrase-based models prefer translation pairs with high frequencies and assign them high

[1] This work has been done while the author was visiting Microsoft Research Asia.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 304–312,
Beijing, August 2010

probability values; yet such pairs could be disliked or even be absent in syntax-based models because of their violation on syntactic restrictions. We alleviate such model bias problem by using the generalized probability features in first-pass decoding, which computed based on feature values from all translation models instead of any single one.

- *Alleviate the over-estimation problem based on translation models with an identical paradigm but different training corpora.*
  In order to obtain further improvements by using an existing training module built for a specified model paradigm, we present a random data sampling method inspired by bagging (Breiman, 1996) to construct translation model ensembles from a unique data set for usage in TMG. Compared to results of TMG based on models with different paradigms, TMG based on models built in such a way can achieve larger improvements.

- *Novel translation probability variance features introduced.*
  We present how to compute the variance for each probability feature based on its values in different involved translation models with prior model probabilities. We add them into the log-linear model as new features to make current SMT models to be more flexible.

The remainder of this paper is organized as follows: we review various translation models in Section 2. In Section 3, we first introduce Bayesian Model Averaging method for SMT tasks and present a generic TMG algorithm based on it. We then discuss two solutions for constructing TM ensembles for usage in TMG. We next introduce probability variance features into current SMT models as new features. We evaluate our method on four state-of-the-art SMT systems, a system combination approach and a model combination approach. Evaluation results are shown in Section 4. In Section 5, we discuss some related work. We conclude the paper in Section 6.

2 Summary of Translation Models

Translation Model (TM) is the most important component in current SMT framework. It provides basic translation units for decoders with a series of probability features for model scoring. Many literatures have paid attentions to TMs from different aspects: DeNeefe *et al.* (2007) compared strengths and weaknesses of a phrase-based TM and a syntax-based TM from the *statistic* aspect; Zollmann *et al.* (2008) made a systematic comparison of three TMs, including phrasal, hierarchical and syntax-based, from the *performance* aspect; and Auli *et al.* (2009) made a systematic analysis of a phrase-based TM and a hierarchical TM from the *search space* aspect.

Given a word-aligned training corpus, we separate a TM training procedure into two phases: *extraction phase* and *parameterization phase*.

Extraction phase aims to pick out all valid translation pairs that are consistent with predefined model constraints. We summarize current TMs based on their corresponding model constraints into two categories below:

- *String-based* TM (string-to-string): reserves all translation pairs that are consistent with word alignment and satisfy length limitation. SMT systems using such TMs can benefit from a large convergence of translation pairs.
- *Tree-based* TM (string-to-tree, tree-to-string or tree-to-tree): needs to obey syntactic restrictions in one side or even both sides of translation candidates. The advantage of using such TMs is that translation outputs trend to be more syntactically well-formed.

Parameterization phase aims to assign a series of probability features to each translation pair. These features play the most important roles in the decision process and are shared by most current SMT decoders. In this paper, we mainly focus on the following four commonly used dominant probability features including:

- translation probability features in two directions: $p(\bar{e}|\bar{f})$ and $p(\bar{f}|\bar{e})$
- lexical weight features in two directions: $p_{lex}(\bar{e}|\bar{f})$ and $p_{lex}(\bar{f}|\bar{e})$

Both string-based and tree-based TMs are state-of-the-art models, and each extraction approach has its own strengths and weaknesses comparing to others. Due to different predefined model constraints, translation pairs extracted by different models usually have different distributions, which could directly affect the resulting probability feature values computed in parameterization phase. In order to utilize translation pairs more fairly in decoding, it is desirable to use more information to measure the quality of translation pairs based on different TMs rather than totally believing any single one.

3 Translation Model Generalization

We first introduce Bayesian Model Averaging method for SMT task. Based on it, we then formally present the generic TMG algorithm. We also provide two solutions for constructing TM ensembles as auxiliary models. We last introduce probability variance features based on multiple TMs for further improvements.

3.1 Bayesian Model Averaging for SMT

Bayesian Model Averaging (BMA) (Hoeting *et al.*, 1999) is a technique designed to solve uncertainty inherent in model selection.

Specifically, for SMT tasks, f is a source sentence, $\mathcal{D}$ is the training data, $\mathcal{M}_k$ is the k^{th} SMT model trained on $\mathcal{D}_k \subset \mathcal{D}$, $p_k(\cdot\,|f,e)$ represents the probability score predicted by $\mathcal{M}_k$ that f can be translated into a target sentence e. BMA provides a way to combine decisions of all $K+1$ SMT models by computing the final translation probability score $\bar{p}_E(\cdot\,|f,e,\mathcal{D})$ as follows:

$$\bar{p}_E(\cdot\,|f,e,\mathcal{D}) = \sum_{k=0}^{K} p(\mathcal{M}_k|\mathcal{D}_k)p_k(\cdot\,|f,e), \quad (1)$$

where $p(\mathcal{M}_k|\mathcal{D}_k)$ is the prior probability that $\mathcal{M}_k$ is a true model. For convenience, we will omit all symbols $\mathcal{D}_k$ in following descriptions.

Ideally, if all involved models $\{\mathcal{M}_0, \dots, \mathcal{M}_K\}$ share the same search space, then translation hypotheses could only be differentiated in probability scores assigned by different SMT models. In such case, BMA can be straightly developed on the whole SMT models in either span level or sentence level to re-compute translation scores of hypotheses for better rankings. However, because of various reasons, e.g. different pruning methods, different training data used, different generative capabilities of SMT models, search spaces between different models are always not identical. Thus, it is intractable to develop BMA on the whole SMT model level directly.

As a tradeoff, we notice that translation pairs between different TMs share a relatively large convergence because of word length limitation. So we instead utilize BMA method to multiple TMs by re-computing values of probability features between them, and we name this process as translation model generalization.

3.2 A Generic BMA-based TMG Algorithm

For a translation model $\mathcal{M}_0$, TMG aims to re-compute its values of probability features based on itself and K collaborative TMs $\{\mathcal{M}_1, \dots, \mathcal{M}_K\}$. We describe the re-computation process for an arbitrary feature $p(\cdot\,|\bar{f},\bar{e}) \in \mathcal{M}_0$ as follows:

$$\bar{p}_E(\cdot\,|\bar{f},\bar{e}) = \sum_{k=0}^{K} p(\mathcal{M}_k)p_k(\cdot\,|\bar{f},\bar{e}), \quad (2)$$

where $p_k(\cdot\,|\bar{f},\bar{e})$ is the feature value assigned by $\mathcal{M}_k$. We denote $\mathcal{M}_0$ as the ***main model***, and other collaborative TMs as ***auxiliary models***. Figure 1 describes an example of TMG on two TMs, where the main model is a phrasal TM.

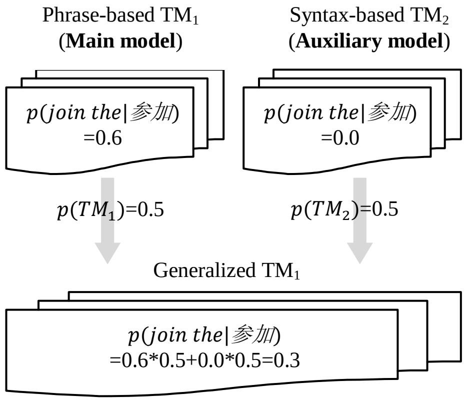


Figure 1. TMG applied to a phrasal TM (main model) and a syntax-based TM (auxiliary model). The value of a translation probability feature $p(join\ the|参加)$ in TM_1 is de-valued (from 0.6 to 0.3), in which 'join the' is absent in TM_2 because of its bad syntactic structure.

Equation 2 is a general framework that can be applied to all TMs. The only limitation is that the segmentation (or tokenization) standards for source (or target) training sentences should be identical for all models. We describe the generic TMG procedure in Algorithm 1[2].

[2] In this paper, since all data sets used have relative large sizes and all SMT models have similar performances, we heuristically set all $p(\mathcal{M}_k)$ equally to $1/(K+1)$.

Algorithm 1: TMG for a main model $\mathcal{M}_0$

for the k^{th} auxiliary TM **do**
    run training procedure on $\mathcal{D}_k$ with specified model constraints and generate $\mathcal{M}_k$
end for
for each translation pair $<\bar{f},\bar{e}>$ in $\mathcal{M}_0$ **do**
    for each probability feature $p(\cdot\,|\bar{f},\bar{e})$ **do**
        for each translation model $\mathcal{M}_k$ **do**
            if $<\bar{f},\bar{e}>$ is contained in $\mathcal{M}_k$ **then**
                $\bar{p}_E(\cdot\,|\bar{f},\bar{e}) \mathrel{+}= p(\mathcal{M}_k)p_k(\cdot\,|\bar{f},\bar{e})$
            end if
        end for
    end for
end for
return the generalized $\mathcal{M}_0$ for SMT decoding

3.3 Auxiliary Model Construction

In order to utilize TMG, more than one TM as auxiliary models is needed. Building TMs with different paradigms is one solution. For example, we can build a syntax-based TM as an auxiliary model for a phrase-based TM. However, it has to re-implement more complicated TM training modules besides the existing one.

In this sub-section, we present an alternative solution to construct auxiliary model ensembles by using the existing training module with different training data extracted from a unique data set. We describe the general procedure for constructing K auxiliary models as follows:

1) Given a unique training corpus $\mathcal{D}$, we randomly sample $N\%$ bilingual sentence pairs without replacement and denote them as $\mathcal{D}_i$. N is a number determined empirically;
2) Based on $\mathcal{D}_i$, we *re-do* word alignment and train an auxiliary model $\mathcal{M}_i$ using the existing training module;
3) We execute Step 1 and Step 2 iteratively for K times, and finally obtain K auxiliary models. The optimal setting of K for TMG is also determined empirically.

With all above steps finished, we can perform TMG as we described in Algorithm 1 based on the K auxiliary models generated already.

The random data sampling process described above is very similar to bagging except for it not allowing replacement during sampling. By making use of this process, translation pairs with low frequencies have relatively high probabilities to be totally discarded, and in resulting TMs, their probabilities could be zero; meanwhile, translation pairs with high frequencies still have high probabilities to be reserved, and hold similar probability feature values in resulting TMs comparing to the main model. Thus, after TMG procedure, feature values could be smoothed for translation pairs with low frequencies, and be stable for translation pairs with high frequencies. From this point of view, TMG can also be seen as a TM smoothing technique based on multiple TMs instead of single one such as Foster *et al.* (2006). We will see in Section 4 that TMG based on TMs generated by both of these two solutions can improve translation quality for all baseline decoders on a series of evaluation sets.

3.4 Probability Variance Feature

The re-computed values of probability features in Equation 2 are actually the feature expectations based on their values from all involved TMs. In order to give more statistical meanings to translation pairs, we also compute their corresponding feature variances based on feature expectations and TM-specified feature values with prior probabilities. We introduce such variances as new features into the log-linear model for further improvements. Our motivation is to quantify the differences of model preferences between TMs for arbitrary probability features.

The variance for an arbitrary probability feature $p(\cdot) \in \mathcal{M}_0$ can be computed as follows:

$$p_V(\cdot) = \sum_{k=0}^{K} \{p_k(\cdot) - \bar{p}_E(\cdot)\}^2 p_k(\mathcal{M}_k), \qquad (3)$$

where $\bar{p}_E(\cdot)$ is the feature expectation computed by Equation 2, $p_k(\cdot)$ is the feature value predicted by $\mathcal{M}_k$, and $p_k(\mathcal{M}_k)$ is the prior probability for $\mathcal{M}_k$. Each probability feature now corresponds to a variance score. We extend the original feature set of $\mathcal{M}_0$ with variance features added in and list the updated set below:

- translation probability expectation features in two directions: $\bar{p}_E(\bar{e}|\bar{f})$ and $\bar{p}_E(\bar{f}|\bar{e})$
- translation probability variance features in two directions: $p_V(\bar{e}|\bar{f})$ and $p_V(\bar{f}|\bar{e})$
- lexical weight expectation features in two directions: $\bar{p}_{E_{lex}}(\bar{e}|\bar{f})$ and $\bar{p}_{E_{lex}}(\bar{f}|\bar{e})$
- lexical weight variance features in two directions: $p_{V_{lex}}(\bar{e}|\bar{f})$ and $p_{V_{lex}}(\bar{f}|\bar{e})$

4 Experiments

4.1 Data Condition

We conduct experiments on the NIST Chinese-to-English MT tasks. We tune model parameters on the NIST 2003 (*MT03*) evaluation set by MERT (Och, 2003), and report results on NIST evaluation sets including the NIST 2004 (*MT04*), the NIST 2005 (*MT05*), the newswire portion of the NIST 2006 (*MT06*) and 2008 (*MT08*). Performances are measured in terms of the case-insensitive BLEU scores in percentage numbers. Table 1 gives statistics over these evaluation sets.

	MT03	MT04	MT05	MT06	MT08
Sent	919	1,788	1,082	616	691
Word	23,788	48,215	29,263	17,316	17,424

Table 1. Statistics on dev/test evaluation sets

We use the *selected data* that picked out from the whole data available for the NIST 2008 constrained track of Chinese-to-English machine translation task as the training corpora, including LDC2003E07, LDC2003E14, LDC2005T06, LDC2005T10, LDC2005E83, LDC2006E26, LDC2006E34, LDC2006E85 and LDC2006E92, which contain about 498,000 sentence pairs after pre-processing. Word alignments are performed by GIZA++ (Och and Ney, 2000) in both directions with an *intersect-diag-grow* refinement.

A traditional 5-gram language model (LM) for all involved systems is trained on the English side of all bilingual data plus the Xinhua portion of LDC English Gigaword Version 3.0. A lexicalized reordering model (Xiong *et al.*, 2006) is trained on the selected data in maximum entropy principle for the phrase-based system. A trigram target dependency LM (DLM) is trained on the English side of the selected data for the dependency-based hierarchical system.

4.2 MT System Description

We include four baseline systems. The first one (***Phr***) is a phrasal system (Xiong *et al.*, 2006) based on Bracketing Transduction Grammar (Wu, 1997) with a lexicalized reordering component based on maximum entropy model. The second one (***Hier***) is a hierarchical phrase-based system (Chiang, 2007) based on Synchronous Context Free Grammar (SCFG). The third one (***Dep***) is a string-to-dependency hierarchical phrase-based system (Shen *et al.*, 2008) with a dependency language model, which translates source strings to target dependency trees. The fourth one (***Synx***) is a syntax-based system (Galley *et al.*, 2006) that translates source strings to target syntactic trees.

4.3 TMG based on Multiple Paradigms

We develop TMG for each baseline system's TM based on the other three TMs as auxiliary models. *All prior probabilities of TMs are set equally to 0.25 heuristically as their similar performances.* Evaluation results are shown in Table 2, where gains more than 0.2 BLEU points are highlighted as improved cases. Compared to baseline systems, systems based on generalized TMs improve in most cases (18 times out of 20). We also notice that the improvements achieved on tree-based systems (Dep and Synx) are relatively smaller than those on string-based systems (Phr and Hier). A potential explanation can be that with considering more syntactic restrictions, tree-based systems suffer less than string-based systems on the over-estimation problem. We do not present further results with variance features added because of their consistent un-promising numbers. *We think this may be due to the considerable portion of non-overlapping translation pairs between main model and auxiliary models, which cause the variances not so accurate*.

		MT03(dev)	MT04	MT05	MT06	MT08	Average
Phr	Baseline	40.45	39.21	38.03	34.24	30.21	36.43
	TMG	41.19**(+0.74)**	39.74**(+0.53)**	38.39**(+0.36)**	34.71**(+0.47)**	30.69**(+0.48)**	36.94**(+0.51)**
Hier	Baseline	41.30	39.63	38.83	34.63	30.46	36.97
	TMG	41.67**(+0.37)**	40.25**(+0.62)**	39.11**(+0.28)**	35.78**(+1.15)**	31.17**(+0.71)**	37.60**(+0.63)**
Dep	Baseline	41.10	39.81	39.47	35.72	30.50	37.32
	TMG	41.37**(+0.27)**	39.92(+0.11)	39.91**(+0.44)**	35.99**(+0.27)**	31.07**(+0.57)**	37.65**(+0.33)**
Synx	Baseline	41.02	39.88	39.47	36.41	32.15	37.79
	TMG	41.26**(+0.24)**	40.09**(+0.21)**	39.90**(+0.43)**	36.77**(+0.36)**	32.15(+0.00)	38.03**(+0.24)**

Table 2. Results of TMG based on TMs with different paradigms

4.4 TMG based on Single Paradigm

We then evaluate TMG based on auxiliary models generated by the random sampling method.

We first decide the percentage of training data to be sampled. We empirically vary this number by 20%, 40%, 60%, 80% and 90% and use each sampled data to train an auxiliary model. We then run TMG on the baseline TM with different auxiliary model used each time. For time saving, we only evaluate on MT03 for Phr in Figure 2.

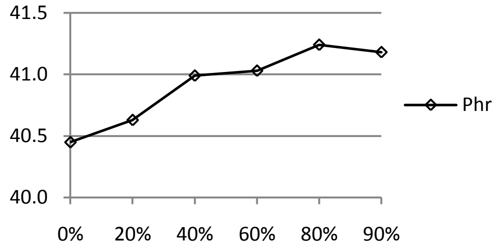


Figure 2. Affects of different percentages of data

The optimal result is achieved when the percentage is 80%, and we fix it as the default value in following experiments.

We then decide the number of auxiliary models used for TMG by varying it from 1 to 5. We list different results on MT03 for Phr in Figure 3.

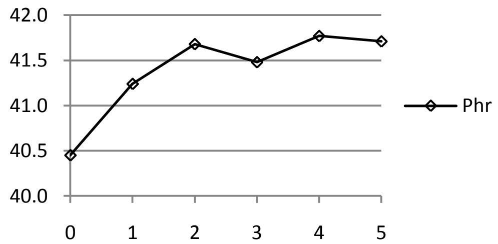


Figure 3. Affects of different numbers of auxiliary models

The optimal result is achieved when the number of auxiliary models is 4, and we fix it as the default value in following experiments.

We now develop TMG for each baseline system's TM based on auxiliary models constructed under default settings determined above. Evaluation results are shown in Table 3. We also investigate the affect of variance features for performance, whose results are denoted as ***TMG+Var***.

From Table 3 we can see that, compared to the results on baseline systems, systems using generalized TMs obtain improvements on almost all evaluation sets (19 times out of 20). With probability variance features added further, the improvements become even more stable than the ones using TMG only (20 times out of 20). Similar to the trend in Table 2, we also notice that TMG method is more preferred by string-based systems (Phr and Hier) rather than tree-based systems (Dep and Synx). This makes our conclusion more solidly that syntactic restrictions can help to alleviate the over-estimation problem.

4.5 Analysis on Phrase Coverage

We next empirically investigate on the translation pair coverage between TM ensembles built by different ways, and use them to analyze results got from previous experiments. Here, we only focus on *full lexicalized* translation entries between models. Those entries with variables are out of consideration in comparisons because of their model dependent properties.

Phrase pairs in the first three TMs have a length limitation in source side up to 3 words, and each source phrase can be translated to at most 20 target phrases.

		MT03(dev)	MT04	MT05	MT06	MT08	Average
	Baseline	40.45	39.21	38.03	34.24	30.21	36.43
Phr	TMG	41.77**(+1.32)**	40.28**(+1.07)**	39.13**(+1.10)**	35.38**(+1.14)**	31.12**(+0.91)**	37.54**(+1.11)**
	TMG+Var	41.77**(+1.32)**	40.31**(+1.10)**	39.43**(+1.30)**	35.61**(+1.37)**	31.62**(+1.41)**	37.74**(+1.31)**
	Baseline	41.30	39.63	38.83	34.63	30.46	36.97
Hier	TMG	42.28**(+0.98)**	40.45**(+0.82)**	39.61**(+0.78)**	35.67**(+1.04)**	31.54**(+1.08)**	37.91**(+0.94)**
	TMG+Var	42.42**(+1.12)**	40.55**(+0.92)**	39.69**(+0.86)**	35.55**(+0.92)**	31.41**(+0.95)**	37.92**(+0.95)**
	Baseline	41.10	39.81	39.47	35.72	30.50	37.32
Dep	TMG	41.49**(+0.39)**	40.20**(+0.39)**	40.00**(+0.53)**	36.13**(+0.41)**	31.24**(+0.74)**	37.81**(+0.49)**
	TMG+Var	41.72**(+0.62)**	40.57**(+0.76)**	40.44**(+0.97)**	36.15**(+0.43)**	31.31**(+0.81)**	38.04**(+0.72)**
	Baseline	41.02	39.88	39.47	36.41	32.15	37.79
Synx	TMG	41.18(+0.16)	40.30**(+0.42)**	39.90**(+0.43)**	36.99**(+0.58)**	32.45**(+0.30)**	38.16**(+0.37)**
	TMG+Var	41.42**(+0.40)**	40.55**(+0.67)**	40.17**(+0.70)**	36.89**(+0.48)**	32.51**(+0.36)**	38.31**(+0.52)**

Table 3. Results of TMG based on TMs constructed by random data sampling

For the fourth TM, these two limitations are released to 4 words and 30 target phrases. We treat phrase pairs identical on both sides but with different syntactic labels in the fourth TM as a unique pair for conveniences in statistics.

We first make statistics on TMs with different paradigms in Table 4. We can see from Table 4 that only slightly over half of the phrase pairs contained by the four involved TMs are common, which is also similar to the conclusion drawn in DeNeefe *et al.* (2006).

Models	#Translation Pair	#Percentage
Phr	1,222,909	**50.6%**
Hier	1,222,909	**50.6%**
Dep	1,087,198	**56.9%**
Synx	1,188,408	**52.0%**
Overlaps	618,371	-

Table 4. Rule statistics on TMs constructed by different paradigms

We then make statistics on TMs with identical paradigm in Table 5. For each baseline TM and its corresponding four auxiliary models constructed by random data sampling, we count the number of phrase pairs that are common between them and compute the percentage numbers based on it for each TM individually.

Models	TM_0	TM_1	TM_2	TM_3	TM_4
Phr	**61.8%**	74.0%	74.1%	73.9%	74.1%
Hier	**61.8%**	74.0%	74.1%	73.9%	74.1%
Dep	**60.8%**	73.6%	73.6%	73.5%	73.7%
Synx	**57.2%**	68.4%	68.5%	68.5%	68.6%

Table 5. Rule statistics on TMs constructed by random sampling (TM_0 is the main model)

Compared to the numbers in Table 4, we find that the coverage between baseline TM and sampled auxiliary models with identical paradigm is larger than that between baseline TM and auxiliary models with different paradigms (about 10 percents). It is a potential reason can explain why results of TMG based on sampled auxiliary models are more effective than those based on auxiliary models built with different paradigms, as we infer that *they share more common phrase pairs each other and make the computation of feature expectations and variances to be more reliable and accurate*.

4.6 Improvements on System Combination

Besides working for single-system decoding, we also perform a system combination method on *N*-best outputs from systems using generalized TMs. We re-implement a state-of-the-art word-level System Combination (**SC**) approach based on incremental HMM alignment proposed by Li *et al.* (2009a). The default number of *N*-best candidates used is set to 20.

We evaluate SC on *N*-best outputs generated from 4 baseline decoders by using different TM settings and list results in Table 6, where ***Base*** stands for combination results on systems using default TMs; ***Paras*** stands for combination results on systems using TMs generalized based on auxiliary models with different paradigms; and ***Samp*** stands for combination results on systems using TMs generalized based on auxiliary models constructed by the random data sampling method. For the Samp setting, we also include probability variance features computed based on Equation 3 in the log-linear model.

SC	MT03	MT04	MT05	MT06	MT08
Base	44.20	42.30	41.22	37.77	33.07
Paras	**44.40**	**42.69**	**41.53**	**38.05**	**33.31**
Samp	**44.80**	**42.95**	**42.10**	**38.39**	**33.67**

Table 6. Results on system combination

From Table 6 we can see that system combination can benefit from TMG method.

4.7 Improvements on Model Combination

As an alternative, model combination is another effective way to improve translation performance by utilizing multiple systems. We re-implement the Model Combination (**MC**) approach (DeNero *et al.*, 2010) using *N*-best lists as its inputs and develop it on *N*-best outputs used in Table 6. Evaluation results are presented in Table 7.

MC	MT03	MT04	MT05	MT06	MT08
Base	42.31	40.57	40.31	38.65	33.88
Paras	**42.87**	**40.96**	**40.77**	**38.81**	**34.47**
Samp	**43.29**	**41.29**	**41.11**	**39.28**	**34.77**

Table 7. Results on model combination

From Table 7 we can see that model combination can also benefit from TMG method.

5 Related Work

Foster and Kuhn (2007) presented an approach that resembles more to our work, in which they divided the training corpus into different components and integrated models trained on each component using the mixture modeling. However, their motivation was to address the *domain adaption problem*, and additional genre information should be provided for the corpus partition to create multiple models for mixture. We instead present two ways for the model ensemble construction without extra information needed: building models by different paradigms or by a random data sampling technique inspired by a machine learning technique. Compared to the prior work, our approach is more general, which can also be used for model adaptation. We can also treat TMG as a smoothing way to address the over-estimation problem existing in almost all TMs. Some literatures have paid attention to this issue as well, such as Foster *et al.* (2006) and Mylonakis and Sima 'an (2008). However, they did not leverage information between multiple models as we did, and developed on single models only. Furthermore, we also make current translation probability features to contain more statistical meanings by introducing the probability variance features into the log-linear model, which are completely novel to prior work and provide further improvements.

6 Conclusion and Future Work

In this paper, we have investigated a simple but effective translation model generalization method that benefits by integrating values of probability features between multiple TMs and using them in decoding phase directly. We also introduce novel probability variance features into the current feature sets of translation models and make the SMT models to be more flexible. We evaluate our method on four state-of-the-art SMT systems, and get promising results not only on single-system decodings, but also on a system combination approach and a model combination approach.

Making use of different distributions of translation probability features is the essential of this work. In the future, we will extend TMG method to other statistical models in SMT framework, (e.g. LM), which could be also suffered from the over-estimation problem. And we will make further research on how to tune prior probabilities of models automatically as well, in order to make our method to be more robust and tunable.

References

Auli Michael, Adam Lopez, Hieu Hoang, and Philipp Koehn. 2009. *A Systematic Analysis of Translation Model Search Spaces*. In *4th Workshop on Statistical Machine Translation*, pages 224-232.

Breiman Leo. 1996. *Bagging Predictors*. *Machine Learning*.

Chiang David. 2007. *Hierarchical Phrase Based Translation*. *Computational Linguistics*, 33(2): 201-228.

DeNero John, Shankar Kumar, Ciprian Chelba, and Franz Och. 2010. *Model Combination for Machine Translation.* To appear in *Proc. of the North American Chapter of the Association for Computational Linguistic.*

DeNeefe Steve, Kevin Knight, Wei Wang, and Daniel Marcu. 2007. *What Can Syntax-based MT Learn from Phrase-based MT?* In *Proc. of Empirical Methods on Natural Language Processing*, pages 755-763.

Foster George, Roland Kuhn, and Howard Johnson. 2006. *Phrasetable Smoothing for Statistical Machine Translation.* In *Proc. of Empirical Methods on Natural Language Processing*, pages 53-61.

Foster George and Roland Kuhn. 2007. *Mixture-Model Adaptation for SMT.* In *2th Workshop on Statistical Machine Translation*, pages 128-135.

Galley Michel, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. *Scalable Inference and Training of Context-Rich Syntactic Translation Models*. In *Proc. of 44th Meeting of the Association for Computational Linguistics*, pages: 961-968.

Huang Liang. 2008. *Forest Reranking: Discriminative Parsing with Non-Local Features*. In *Proc. of 46th Meeting of the Association for Computational Linguistics*, pages 586-594.

Hoeting Jennifer, David Madigan, Adrian Raftery, and Chris Volinsky. 1999. *Bayesian Model Averaging: A tutorial. Statistical Science*, Vol. 14, pages 382-417.

He Xiaodong, Mei Yang, Jianfeng Gao, Patrick Nguyen, and Robert Moore. 2008. *Indirect-HMM-based Hypothesis Alignment for Combining Outputs from Machine Translation Systems*. In *Proc. of Empirical Methods on Natural Language Processing*, pages 98-107.

Koehn Philipp. 2004. *Phrase-based Model for SMT. Computational Linguistics*, 28(1): 114-133.

Li Chi-Ho, Xiaodong He, Yupeng Liu, and Ning Xi. 2009a. *Incremental HMM Alignment for MT system Combination*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 949-957.

Li Mu, Nan Duan, Dongdong Zhang, Chi-Ho Li, and Ming Zhou. 2009b. *Collaborative Decoding: Partial Hypothesis Re-Ranking Using Translation Consensus between Decoders*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 585-592.

Liu Yang, Haitao Mi, Yang Feng, and Qun Liu. 2009. *Joint Decoding with Multiple Translation Models*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 576-584.

Mylonakis Markos and Khalil Sima 'an. 2008. *Phrase Translation Probabilities with ITG Priors and Smoothing as Learning Objective*. In *Proc. of Empirical Methods on Natural Language Processing*, pages 630-639.

Matusov Evgeny, Nicola Ueffi ng, and Hermann Ney. 2006. Computing consensus translation from multiple machine translation systems using enhanced hypotheses alignment. In *Proc. of European Charter of the Association for Computational Linguistics*, pages 33-40.

Och Franz and Hermann Ney. 2000. *Improved Statistical Alignment Models*. In *Proc. of 38th Meeting of the Association for Computational Linguistics*, pages 440-447.

Och Franz. 2003. *Minimum Error Rate Training in Statistical Machine Translation*. In *Proc. of 41th Meeting of the Association for Computational Linguistics*, pages 160-167.

Och Franz and Hermann Ney. 2004. *The Alignment template approach to Statistical Machine Translation*. *Computational Linguistics*, 30(4): 417-449.

Shen Libin, Jinxi Xu, and Ralph Weischedel. 2008. *A new string-to-dependency machine translation algorithm with a target dependency language model*. In *Proc. of 46th Meeting of the Association for Computational Linguistics*, pages 577-585.

Wu Dekai. 1997. *Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora*. *Computational Linguistics*, 23(3): 377-404.

Xiong Deyi, Qun Liu, and Shouxun Lin. 2006. *Maximum Entropy based Phrase Reordering Model for Statistical Machine Translation*. In *Proc. of 44th Meeting of the Association for Computational Linguistics*, pages 521-528.

Zollmann Andreas, Ashish Venugopal, Franz Och, and Jay Ponte. 2008. *A Systematic Comparison of Phrase-Based, Hierarchical and Syntax-Augmented Statistical MT.* In *23rd International Conference on Computational Linguistics*, pages 1145-1152.

Mixture Model-based Minimum Bayes Risk Decoding using Multiple Machine Translation Systems

Nan Duan[1]
School of Computer Science and Technology
Tianjin University
v-naduan@microsoft.com

Mu Li, Dongdong Zhang, Ming Zhou
Microsoft Research Asia
muli@microsoft.com
dozhang@microsoft.com
mingzhou@microsoft.com

Abstract

We present Mixture Model-based Minimum Bayes Risk (MMMBR) decoding, an approach that makes use of multiple SMT systems to improve translation accuracy. Unlike existing MBR decoding methods defined on the basis of single SMT systems, an MMMBR decoder reranks translation outputs in the combined search space of multiple systems using the MBR decision rule and a mixture distribution of component SMT models for translation hypotheses. MMMBR decoding is a general method that is independent of specific SMT models and can be applied to various commonly used search spaces. Experimental results on the NIST Chinese-to-English MT evaluation tasks show that our approach brings significant improvements to single system-based MBR decoding and outperforms a state-of-the-art system combination method.

1 Introduction

Minimum Bayes Risk (MBR) decoding is becoming more and more popular in recent Statistical Machine Translation (SMT) research. This approach requires a second-pass decoding procedure to re-rank translation hypotheses by risk scores computed based on model's distribution.

Kumar and Byrne (2004) first introduced MBR decoding to SMT field and developed it on the *N*-best list translations. Their work has shown that MBR decoding performs better than Maximum a Posteriori (MAP) decoding for different evaluation criteria. After that, many dedicated efforts have been made to improve the performances of SMT systems by utilizing MBR-inspired methods. Tromble *et al.* (2008) proposed a linear approximation to BLEU score (log-BLEU) as a new loss function in MBR decoding and extended it from *N*-best lists to lattices, and Kumar *et al.* (2009) presented more efficient algorithms for MBR decoding on both lattices and hypergraphs to alleviate the high computational cost problem in Tromble *et al.*'s work. DeNero *et al.* (2009) proposed a fast consensus decoding algorithm for MBR for both linear and non-linear similarity measures.

All work mentioned above share a common setting: an MBR decoder is built based on one and only one MAP decoder. On the other hand, recent research has shown that substantial improvements can be achieved by utilizing consensus statistics over multiple SMT systems (Rosti *et al.*, 2007; Li *et al.*, 2009a; Li *et al.*, 2009b; Liu *et al.*, 2009). It could be desirable to adapt MBR decoding to multiple SMT systems as well.

In this paper, we present *Mixture Model-based Minimum Bayes Risk (MMMBR) decoding*, an approach that makes use of multiple SMT systems to improve translation performance. In this work, we can take advantage of a larger search space for hypothesis selection, and employ an improved probability distribution over translation hypotheses based on mixture modeling, which linearly combines distributions of multiple component systems for Bayes risk computation. The key contribution of this paper is the usage of mixture modeling in MBR, which allows multiple SMT models to be involved in and makes the computation of *n*-gram consensus statistics to be more accurate. Evaluation results have shown that our approach not only brings significant improvements to single system-based MBR decoding but also outperforms a state-of-the-art word-level system combination method.

[1] This work has been done while the author was visiting Microsoft Research Asia.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 313–321,
Beijing, August 2010

The rest of the paper is organized as follows: In Section 2, we first review traditional MBR decoding method and summarize various search spaces that can be utilized by an MBR decoder. Then, we describe how a mixture model can be used to combine distributions of multiple SMT systems for Bayes risk computation. Lastly, we present detailed MMMBR decoding model on multiple systems and make comparison with single system-based MBR decoding methods. Section 3 describes how to optimize different types of parameters. Experimental results will be shown in Section 4. Section 5 discusses some related work and Section 6 concludes the paper.

2 Mixture Model-based MBR Decoding

2.1 Minimum Bayes Risk Decoding

Given a source sentence F, MBR decoding aims to find the translation with the least expected loss under a probability distribution. The objective of an MBR decoder can be written as:

$$\hat{E} = \underset{E' \in \mathcal{H}_h}{\operatorname{argmin}} \sum_{E \in \mathcal{H}_e} L(E, E') P(E|F, \mathcal{H}_e). \quad (1)$$

where $\mathcal{H}_h$ denotes a *search space* for hypothesis selection; $\mathcal{H}_e$ denotes an *evidence space* for Bayes risk computation; $L(\cdot)$ denotes a function that measures the loss between E' and E; $P(\cdot)$ is the underlying distribution based on $\mathcal{H}_e$.

Some of existing work on MBR decoding focused on exploring larger spaces for both $\mathcal{H}_h$ and $\mathcal{H}_e$, e.g. from *N*-best lists to lattices or hypergraphs (Tromble *et al.*, 2008; Kumar *et al.*, 2009). Various loss functions have also been investigated by using different evaluation criteria for similarity computation, e.g. Word Error Rate, Position-independent Word Error Rate, BLEU and log-BLEU (Kumar and Byrne, 2004; Tromble *et al.*, 2008). But less attention has been paid to distribution $P(\cdot)$. Currently, many SMT systems based on different paradigms can yield similar performances but are good at modeling different inputs in the translation task (Koehn *et al.*, 2004a; Och *et al.*, 2004; Chiang, 2007; Mi *et al.*, 2008; Huang, 2008). We expect to integrate the advantages of different SMT models into MBR decoding for further improvements. In particular, we make in-depth investigation into MBR decoding concentrating on the translation distribution $P(\cdot)$ by leveraging a mixture model based on multiple SMT systems.

2.2 Summary of Translation Search Spaces

There are three major forms of search spaces that can be obtained from an MAP decoder as a byproduct, depending on the design of the decoder: *N*-best lists, lattices and hypergraphs.

An *N*-best list contains the *N* most probable translation hypotheses produced by a decoder. It only presents a very small portion of the entire search space of an SMT model.

A hypergraph is a weighted acyclic graph which compactly encodes an exponential number of translation hypotheses. It allows us to represent both phrase-based and syntax-based systems in a unified framework. Formally, a hypergraph $\mathcal{H}$ is a pair $< \mathcal{V}, \mathcal{E} >$, where $\mathcal{V}$ is a set of hypernodes and $\mathcal{E}$ is a set of hyperedges. Each hypernode $v \in \mathcal{V}$ corresponds to translation hypotheses with identical decoding states, which usually include the span (i, j) of the words being translated, the grammar symbol s for that span and the left and right boundary words of hypotheses for computing language model (LM) scores. Each hyperedge $e \in \mathcal{E}$ corresponds to a translation rule and connects a head node $h(e)$ and a set of tail nodes $T(e)$. The number of tail nodes $|T(e)|$ is called the *arity* of the hyperedge *e* and the arity of a hypergraph is the maximum arity of its hyperedges. If the arity of a hyperedge *e* is zero, $h(e)$ is then called a *source node*. Each hypergraph has a unique *root node* and each path in a hypergraph induces a translation hypothesis. A lattice (Ueffing *et al.*, 2002) can be viewed as a special hypergraph, in which the maximum arity is one.

2.3 Mixture Model for SMT

We first describe how to construct a general distribution for translation hypotheses over multiple SMT systems using mixture modeling for usage in MBR decoding.

Mixture modeling is a technique that has been applied to many statistical tasks successfully. For the SMT task in particular, given K SMT systems with their corresponding model distributions, a mixture model is defined as a probability distribution over the combined search space of all component systems and computed as a weighted sum of component model distributions:

$$P(E|F,\mathcal{H}) = \sum_{k=1}^{K} \lambda_k P_k(E|F,\mathcal{H}_k). \qquad (2)$$

In Equation 2, $\lambda_1, \lambda_2, \ldots, \lambda_K$ are system weights which hold following constraints: $0 \le \lambda_k \le 1$ and $\sum_{k=1}^{K} \lambda_k = 1$, $P_k(E|F,\mathcal{H}_k)$ is the k^{th} distribution estimated on the search space $\mathcal{H}_k$ based on the log-linear formulation:

$$P_k(E|F,\mathcal{H}_k) = \frac{exp(\alpha_k \theta_k(E,F))}{\sum_{E' \in \mathcal{H}_k} exp(\alpha_k \theta_k(E',F))},$$

where $\theta_k(E,F)$ is the score function of the k^{th} system for translation E, $\alpha_k \in [0,\infty)$ is a scaling factor that determines the flatness of the distribution P_k sharp ($\alpha_k > 1$) or smooth ($\alpha_k < 1$).

Due to the inherent differences in SMT models, translation hypotheses have different distributions in different systems. A mixture model can effectively combine multiple distributions with tunable system weights. The distribution of a single model used in traditional MBR can be seen as a special mixture model, where K is one.

2.4 Mixture Model for SMT

Let $\{d_1, d_2, \ldots, d_K\}$ denote K machine translation systems, $\mathcal{H}_i$ denotes the search space produced by system d_i in MAP decoding procedure. An MMMBR decoder aims to seek a translation from the combined search space $\mathcal{H} = \bigcup_i \mathcal{H}_i$ that maximizes the expected gain score based on a mixture model $P(E|F,\mathcal{H})$. We write the objective function of MMMBR decoding as:

$$\hat{E} = \underset{E' \in \mathcal{H}}{\operatorname{argmax}} \sum_{E \in \mathcal{H}} G(E,E') P(E|F,\mathcal{H}). \qquad (3)$$

For the gain function $G(\cdot)$, we follow Tromble *et al.* (2008) to use log-BLEU, which is scored by the hypothesis length and a linear function of *n*-gram matches as:

$$G(E,E') = \theta_0 |E'| + \sum_{\omega} \theta_{|\omega|} \#_{\omega}(E') \delta_{\omega}(E),$$

In this definition, E is a reference translation, $|E'|$ is the length of hypothesis E', ω is an *n*-gram presented in E', $\#_{\omega}(E')$ is the number of times that ω occurs in E', and $\delta_{\omega}(E)$ is an indicator function which equals to 1 when ω occurs in E and 0 otherwise. $\theta_0, \theta_1, \ldots, \theta_N$ are model parameters, where N is the maximum order of the *n*-grams involved.

For the mixture model $P(\cdot)$, we replace it by Equation 2 and rewrite the total gain score for hypothesis E' in Equation 3:

$$\begin{aligned}
&\sum_{E \in \mathcal{H}} G(E,E') P(E|F,\mathcal{H}) \\
&= \sum_{E \in \mathcal{H}} G(E,E') \sum_{i} \lambda_i P_i(E|F,\mathcal{H}_i) \\
&= \sum_{i} \lambda_i \sum_{E \in \mathcal{H}} G(E,E') P_i(E|F,\mathcal{H}_i) \\
&= \sum_{i} \lambda_i \sum_{k=1}^{K} \sum_{E \in \mathcal{H}_k} G(E,E') P_i(E|F,\mathcal{H}_i).
\end{aligned} \qquad (4)$$

In Equation 4, the total gain score on the combined search space $\mathcal{H}$ can be further decomposed into each local search space $\mathcal{H}_k$ with a specified distribution $P_i(E|F,\mathcal{H}_i)$. This is a nice property and it allows us to compute the total gain score as a weighted sum of local gain scores on different search spaces. We expand the local gain score for E' computed on search space $\mathcal{H}_k$ with $P_i(E|F,\mathcal{H}_i)$ using log-BLEU as:

$$\begin{aligned}
&\sum_{E \in \mathcal{H}_k} G(E,E') P_i(E|F,\mathcal{H}_i) \\
&= \sum_{E \in \mathcal{H}_k} \left\{ \theta_0 |E'| + \sum_{\omega} \theta_{|\omega|} \#_{\omega}(E') \delta_{\omega}(E) \right\} P_i(E|F,\mathcal{H}_i) \\
&\approx \theta_0 |E'| + \sum_{\omega} \theta_{|\omega|} \#_{\omega}(E') p_i(\omega|\mathcal{H}_i).
\end{aligned} \qquad (5)$$

We make two approximations for the situations when $i \ne k$: the first is $\sum_{E \in \mathcal{H}_k} P_i(E|F,\mathcal{H}_i) \approx 1$ and the second is $\sum_{E \in \mathcal{H}_k} \delta_{\omega}(E) P_i(E|F,\mathcal{H}_i) \approx p_i(\omega|\mathcal{H}_i)$. In fact, due to the differences in generative capabilities of SMT models, training data selection and various pruning techniques used, search spaces of different systems are always not identical in practice. For the convenience of formal analysis, we treat all $P_i(E|F,\mathcal{H}_i)$ as ideal distributions with assumptions that all systems work in similar settings, and translation candidates are shared by all systems.

The method for computing *n*-gram posterior probability $p_i(\omega|\mathcal{H}_i)$ in Equation 5 depends on different types of search space $\mathcal{H}_i$:

- When $\mathcal{H}_i$ is an *N*-best list, it can be computed immediately by enumerating all translation candidates in the *N*-best list:

$$p_i(\omega|\mathcal{H}_i) = \sum_{E \in \mathcal{H}_i} \delta_{\omega}(E) P_i(E|F,\mathcal{H}_i).$$

- When $\mathcal{H}_i$ is a hypergraph (or a lattice) that encodes exponential number of hypotheses, it is often impractical to compute this probability directly. In this paper, we use the algorithm presented in Kumar *et al.* (2009) which is described in Algorithm 1[2]:

$$p_i(\omega|\mathcal{H}_i) = \sum_{E\in\mathcal{H}_i}\sum_{e\in E} f^*(e,\omega,\mathcal{H}_i)\, P_i(E|F,\mathcal{H}_i)$$
$$= \sum_{e\in\mathcal{E}} 1_e(\omega) f^*(e,\omega,\mathcal{H}_i) \sum_{E\in\mathcal{H}_i} 1_E(e) P_i(E|F,\mathcal{H}_i)$$
$$= \sum_{e\in\mathcal{E}} 1_e(\omega) f^*(e,\omega,\mathcal{H}_i) p_i(e|\mathcal{H}_i).$$

$f^*(e,\omega,\mathcal{H}_i)$ counts the edge e with *n*-gram ω that has the highest edge posterior probability relative to predecessors in the entire graph $\mathcal{H}_i$, and $p_i(e|\mathcal{H}_i)$ is the edge posterior probability that can be efficiently computed with standard inside and outside probabilities $I(v)$ and $O(v)$ as:

$$p_i(e|\mathcal{H}_i) = \frac{1}{Z(f)} \omega(e) O\big(h(e)\big) \prod_{v\in T(e)} I(v),$$

where $\omega(e)$ is the weight of hyperedge e in $\mathcal{H}_i$, $Z(f)$ is the normalization factor that equals to the inside probability of the root node in $\mathcal{H}_i$.

Algorithm 1: Compute *n*-gram posterior probabilities on hypergraph $\mathcal{H}_i$ (Kumar *et al.*, 2009)

```
sort hypernodes topologically
compute inside/outside probabilities I(v) and O(v)
  for each hypernode v ∈ H_i
compute edge posterior probability p_i(e|H_i) for
  each hyperedge e ∈ H_i
for each hyperedge e ∈ H_i do
    merge n-grams on T(e) and keep the highest
    probability when n-grams are duplicated
    apply the rule of edge e to n-grams on T(e) and
    propagate n − 1 gram prefixes/suffixes to h(e)
    for each n-gram ω introduced by e do
        if p_i(e|H_i) > Max(ω, T(e)) then
            p_i(ω|H_i) += p_i(e|H_i) − Max(ω, T(e))
            Max(ω, h(e)) = p_i(e|H_i)
        else
            Max(ω, h(e)) = Max(ω, T(e))
        end if
    end for
end for
return n-gram posterior probability set {p_i(ω|H_i)}_ω
```

[2] We omit the similar algorithm for lattices because of their homogenous structures comparing to hypergraphs as we discussed in Section 2.2.

Thus, the total gain score for hypothesis E' on $\mathcal{H} = \bigcup_i \mathcal{H}_i$ can be further expanded as:

$$\sum_i \lambda_i \sum_{k=1}^{K} \sum_{E\in\mathcal{H}_k} G(E,E') P_i(E|F,\mathcal{H}_i)$$
$$\approx \sum_i \lambda_i \sum_{k=1}^{K} \left\{\theta_0|E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') p_i(\omega|\mathcal{H}_i)\right\}$$
$$= \sum_i \lambda_i K \left\{\theta_0|E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') p_i(\omega|\mathcal{H}_i)\right\}$$
$$= K\left\{\sum_i \lambda_i \theta_0 |E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') \sum_i \lambda_i p_i(\omega|\mathcal{H}_i)\right\}$$
$$= K\left\{\theta_0|E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') \wp(\omega)\right\} \quad (6)$$

where $\wp(\omega) = \sum_i \lambda_i p_i(\omega|\mathcal{H}_i)$ is a mixture *n*-gram posterior probability. The most important fact derived from Equation 6 is that, the mixture of different distributions can be simplified to the weighted sum of *n*-gram posterior probabilities on different search spaces.

We now derive the decision rule of MMMBR decoding based on Equation 6 below:

$$\hat{E} = \underset{E'\in\mathcal{H}}{\operatorname{argmax}} \theta_0|E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') \wp(\omega). \quad (7)$$

We also notice that MAP decoding and MBR decoding are two different ways of estimating the probability $P(E|F)$ and each of them has advantages and disadvantages. It is desirable to interpolate them together when choosing the final translation outputs. So we include each system's MAP decoding cost as an additional feature further and modify Equation 7 to:

$$\hat{E} = \underset{E'\in\mathcal{H}}{\operatorname{argmax}} \theta_0|E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') \wp(\omega) + \sum_k \theta_k \log C_{MAP}(E'|F,d_k), \quad (8)$$

where $C_{MAP}(E'|F,d_k)$ is the model cost assigned by the MAP decoder d_k for hypothesis E'. Because the costs of MAP decoding on different SMT models are not directly comparable, we utilize the MERT algorithm to assign an appropriate weight θ_k for each component system.

Compared to single system-based MBR decoding, which obeys the decision rule below:

$$\hat{E} = \underset{E'\in\mathcal{H}_k}{\operatorname{argmax}} \theta_0|E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') p(\omega|\mathcal{H}_k),$$

MMMBR decoding has a similar objective function (Equation 8). The key difference is that, in MMMBR decoding, n-gram posterior probability $\wp(\omega)$ is computed as $\sum_i \lambda_i\, p_i(\omega|\mathcal{H}_i)$ based on an ensemble of search spaces; meanwhile, in single system-based MBR decoding, this quantity is computed locally on single search space $\mathcal{H}_k$. The procedure of MMMBR decoding on multiple SMT systems is described in Algorithm 2.

Algorithm 2: MMMBR decoding on multiple SMT systems

```
1:  for each component system d_k do
2:      run MAP decoding and generate the corresponding search space H_k
3:      compute the n-gram posterior probability set {p_k(ω|H_k)}_ω for H_k based on Algorithm 1
4:  end for
5   compute the mixture n-gram posterior probability ℘(ω) = Σ_i λ_i p_i(ω|H_i) for each ω:
6:  for each unique n-gram ω appeared in ∪_k H_k do
7:      for each search space H_i do
8           ℘(ω) += λ_i p_i(ω|H_i)
9:      end for
10: end for
11: for each hyperedge e in ∪_k H_k do
12:     assign ℘(ω) to the edge e for all ω contained in e
13: end for
14: return the best path according to Equation 8
```

3 A Two-Pass Parameter Optimization

In Equation 8, there are two types of parameters: parameters introduced by the gain function $G(\cdot)$ and the model cost $C_{MAP}(\cdot)$, and system weights introduced by the mixture model $P(\cdot)$. Because Equation 8 is not a linear function when all parameters are taken into account, MERT algorithm (Och, 2003) cannot be directly applied to optimize them at the same time. Our solution is to employ a two-pass training strategy, in which we optimize parameters for MBR first and then system weights for the mixture model.

3.1 Parameter Optimization for MBR

The inputs of an MMMBR decoder can be a combination of translation search spaces with arbitrary structures. For the sake of a general and convenience solution for optimization, we utilize the simplest N-best lists with proper sizes as approximations to arbitrary search spaces to optimize MBR parameters using MERT in the first-pass training. System weights can be set empirically based on different performances, or equally without any bias. Note that although we tune MBR parameters on N-best lists, n-gram posterior probabilities used for Bayes risk computation could still be estimated on hypergraphs for non N-best-based search spaces.

3.2 Parameter Optimization for Mixture Model

After MBR parameters optimized, we begin to tune system weights for the mixture model in the second-pass training. We rewrite Equation 8 as:

$$\hat{E} = \underset{E' \in \mathcal{H}}{\operatorname{argmax}} \sum_i \lambda_i \{\theta_0 |E'| + \sum_\omega \theta_{|\omega|} \#_\omega(E') p_i(\omega|\mathcal{H}_i) + \sum_k \theta_k \log C_{MAP}(E'|F, d_k)\}. \quad (9)$$

For each λ_i, the aggregated score surrounded with braces can be seen as its feature value. Equation 9 now turns to be a linear function for all weights and can be optimized by the MERT.

4 Experiments

4.1 Data and Metric

We conduct experiments on the NIST Chinese-to-English machine translation tasks. We use the newswire portion of the NIST 2006 test set (*MT06-nw*) as the development set for parameter optimization, and report results on the NIST 2008 test set (*MT08*). Translation performances are measured in terms of case-insensitive BLEU scores. Statistical significance is computed using the bootstrap re-sampling method proposed by Koehn (2004b). Table 1 gives data statistics.

Data Set	#Sentence	#Word
MT06-nw (dev)	616	17,316
MT08 (test)	1,357	31,600

Table 1. Statistics on dev and test data sets

All bilingual corpora available for the NIST 2008 constrained track of Chinese-to-English machine translation task are used as training data, which contain 5.1M sentence pairs, 128M Chinese words and 147M English words after pre-processing. Word alignments are performed by GIZA++ with an intersect-diag-grow refinement.

A 5-gram language model is trained on the English side of all bilingual data plus the Xinhua portion of LDC English Gigaword Version 3.0.

4.2 System Description

We use two baseline systems. The first one (*SYS1*) is a hierarchical phrase-based system (Chiang, 2007) based on Synchronous Context Free Grammar (SCFG), and the second one (*SYS2*) is a phrasal system (Xiong *et al.*, 2006) based on Bracketing Transduction Grammar (Wu, 1997) with a lexicalized reordering component based on maximum entropy model. Phrasal rules shared by both systems are extracted on all bilingual data, while hierarchical rules for SYS1 only are extracted on a selected data set, including LDC2003E07, LDC2003E14, LDC2005T06, LDC2005T10, LDC2005E83, LDC2006E26, LDC2006E34, LDC2006E85 and LDC2006E92, which contain about 498,000 sentence pairs. Translation hypergraphs are generated by each baseline system during the MAP decoding phase, and 1000-best lists used for MERT algorithm are extracted from hypergraphs by the *k*-best parsing algorithm (Huang and Chiang, 2005). We tune scaling factor to optimize the performance of HyperGraph-based MBR decoding (HGMBR) on MT06-nw for each system (0.5 for SYS1 and 0.01 for SYS2).

4.3 MMMBR Results on Multiple Systems

We first present the overall results of MMMBR decoding on two baseline systems.

To compare with single system-based MBR methods, we re-implement *N-best MBR*, which performs MBR decoding on 1000-best lists with the fast consensus decoding algorithm (DeNero *et al.*, 2009), and *HGMBR*, which performs MBR decoding on a hypergraph (Kumar *et al.*, 2009). Both methods use log-BLEU as the loss function. We also compare our method with *IHMM Word-Comb*, a state-of-the-art word-level system combination approach based on incremental HMM alignment proposed by Li *et al.* (2009b). We report results of MMMBR decoding on both *N*-best lists (*N-best MMMBR*) and hypergraphs (*Hypergraph MMMBR*) of two baseline systems. As MBR decoding can be used for any SMT system, we also evaluate *MBR-IHMM Word-Comb*, which uses *N*-best lists generated by HGMBR on each baseline systems.

The default beam size is set to 50 for MAP decoding and hypergraph generation. The setting of *N*-best candidates used for (MBR-) IHMM Word-Comb is the same as the one used in Li *et al.* (2009b). The maximum order of *n*-grams involved in MBR model is set to 4. Table 2 shows the evaluation results.

	MT06-nw		MT08	
	SYS1	SYS2	SYS1	SYS2
MAP	38.1	37.1	28.5	28.0
N-best MBR	38.3	37.4	29.0	28.1
HGMBR	38.3	37.5	29.1	28.3
IHMM Word-Comb	39.1		29.3	
MBR-IHMM Word-Comb	39.3		29.7	
N-best MMMBR	39.0*		29.4*	
Hypergraph MMMBR	**39.4***$^{+}$		**29.9***$^{+}$	

Table 2. MMMBR decoding on multiple systems (*: significantly better than HGMBR with $p < 0.01$; +: significantly better than IHMM Word-Comb with $p < 0.05$)

From Table 2 we can see that, compared to MAP decoding, *N*-best MBR and HGMBR only improve the performance in a relative small range (+0.1~+0.6 BLEU), while MMMBR decoding on multiple systems can yield significant improvements on both dev set (+0.9 BLEU on *N*-best MMMBR and +1.3 BLEU on Hypergraph MMMBR) and test set (+0.9 BLEU on *N*-best MMMBR and +1.4 BLEU on Hypergraph MMMBR); compared to IHMM Word-Comb, *N*-best MMMBR can achieve comparable results on both dev and test sets, while Hypergraphs MMMBR can achieve even better results (+0.3 BLEU on dev and +0.6 BLEU on test); compared to MBR-IHMM Word-Comb, Hypergraph MMMBR can also obtain comparable results with tiny improvements (+0.1 BLEU on dev and +0.2 BLEU on test). However, MBR-IHMM Word-Comb has ability to generate new hypotheses, while Hypergraph MMMBR only chooses translations from original search spaces.

We next evaluate performances of MMMBR decoding on hypergraphs generated by different beam size settings, and compare them to (MBR-)

IHMM Word-Comb with the same candidate size and HGMBR with the same beam size. We list the results of MAP decoding for comparison. The comparative results on MT08 are shown in Figure 1, ***where X-axis is the size used for all methods each time***, ***Y-axis is the BLEU score***, MAP-i and HGMBR-i stand for MAP decoding and HGMBR decoding for the i^{th} system.

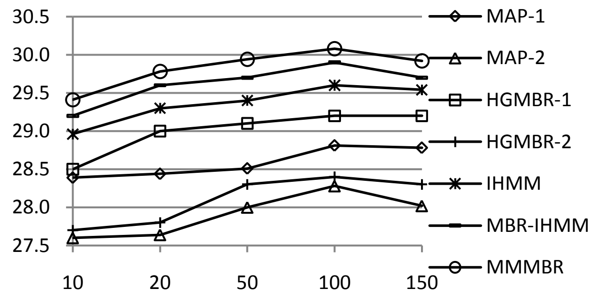


Figure 1. MMMBR vs. (MBR-) IHMM Word-Comb and HGMBR with different sizes

From Figure 1 we can see that, MMMBR decoding performs consistently better than both (MBR-) IHMM Word-Comb and HGMBR on all sizes. The gains achieved are around +0.5 BLEU compared to IHMM Word-Comb, +0.2 BLEU compared to MBR-IHMM Word-Comb, and +0.8 BLEU compared to HGMBR. Compared to MAP decoding, the best result (30.1) is obtained when the size is 100, and the largest improvement (+1.4 BLEU) is obtained when the size is 50. However, we did not observe significant improvement when the size is larger than 50.

We then setup an experiment to verify that the mixture model based on multiple distributions is more effective than any individual distributions for Bayes risk computation in MBR decoding. We use Mix-HGMBR to denote MBR decoding performed on single hypergraph of each system in the meantime using a mixture model upon distributions of two systems for Bayes risk computation. We compare it with HGMBR and Hypergraph MMMBR and list results in Table 3.

	MT08	
	SYS1	SYS2
HGMBR	29.1	28.3
Mix-HGMBR	29.4	28.9
Hypergraph MMMBR	**29.9**	

Table 3. Performance of MBR decoding on different settings of search spaces and distributions

It can be seen that based on the same search space, the performance of Mix-HGMBR is significantly better than that of HGMBR (+0.3/+0.6 BLEU on dev/test). Yet the performance is still not as good as Hypergraph, which indicates the fact that the mixture model and the combination of search spaces are both helpful to MBR decoding, and the best choice is to use them together.

We also empirically investigate the impacts of different system weight settings upon the performances of Hypergraph MMMBR on dev set in Figure 2, ***where X-axis is the weight λ_1 for SYS1***, ***Y-axis is the BLEU score***. The weight λ_2 for SYS2 equals to $1 - \lambda_1$ as only two systems involved. The best evaluation result on dev set is achieved when the weight pair is set to 0.7/0.3 for SYS1/SYS2, which is also very close to the one trained automatically by the training strategy presented in Section 3.2. Although this training strategy can be processed repeatedly, the performance is stable after the 1st round finished.

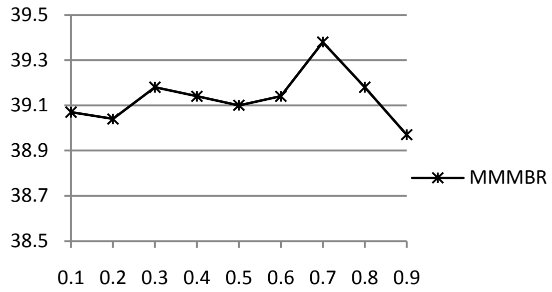


Figure 2. Impacts of different system weights in the mixture model

4.4 MMMBR Results on Identical Systems with Different Translation Models

Inspired by Macherey and Och (2007), we arrange a similar experiment to test MMMBR decoding for each baseline system on an ensemble of sub-systems built by the following two steps.

Firstly, we iteratively apply the following procedure 3 times: at the i^{th} time, we randomly sample 80% sentence pairs from the total bilingual data to train a translation model and use it to build a new system based on the same decoder, which is denoted as *sub-system-i*. Table 4 shows the evaluation results of all sub-systems on MT08, where MAP decoding (the former ones) and corresponding HGMBR (the latter ones) are grouped together by a slash. We set all beam sizes to 20 for a time-saving purpose.

	MT08	
	SYS1	SYS2
Baseline	28.4/29.0	27.6/27.8
sub-system-1	28.1/28.5	26.8/27.3
sub-system-2	28.3/28.4	27.0/27.1
sub-system-3	27.7/28.0	27.3/27.6

Table 4. Performance of sub-systems

Secondly, starting from each baseline system, we gradually add one more sub-system each time and perform Hypergraph MMMBR on hypergraphs generated by current involved systems. Table 5 shows the evaluation results.

	MT08	
	SYS1	SYS2
MAP	28.4	27.6
HGMBR	29.0	27.8
Hypergraph MMMBR		
+ sub-system-1	29.1	27.9
+ sub-system-2	29.1	28.1
+ sub-system-3	**29.3**	**28.3**

Table 5. Performance of Hypergraph MMMBR on multiple sub-systems

We can see from Table 5 that, compared to the results of MAP decoding, MMMBR decoding can achieve significant improvements when more than one sub-system are involved; however, compared to the results of HGMBR on baseline systems, there are few changes of performance when the number of sub-systems increases. One potential reason is that the translation hypotheses between multiple sub-systems under the same SMT model hold high degree of correlation, which is discussed in Macherey and Och (2007).

We also evaluate MBR-IHMM Word-Comb on *N*-best lists generated by each baseline system with its corresponding three sub-systems. Evaluation results are shown in Table 6, where Hypergraph MMMBR still outperforms MBR-IHMM Word-Comb on both baseline systems.

	MT08	
	SYS1	SYS2
MBR-IHMM Word-Comb	29.1	28.0
Hypergraph MMMBR	**29.3**	**28.3**

Table 6. Hypergraph MMMBR vs. MBR-IHMM Word-Comb with multiple sub-systems

5 Related Work

Employing consensus between multiple systems to improve machine translation quality has made rapid progress in recent years. System combination methods based on confusion networks (Rosti *et al.*, 2007; Li *et al.*, 2009b) have shown state-of-the-art performances in MT benchmarks. Different from them, MMMBR decoding method does not generate new translations. It maintains the essential of MBR methods to seek translations from existing search spaces. Hypothesis selection method (Hildebrand and Vogel, 2008) resembles more our method in making use of *n*-gram statistics. Yet their work does not belong to the MBR framework and treats all systems equally. Li *et al.* (2009a) presents a co-decoding method, in which *n*-gram agreement and disagreement statistics between translations of multiple decoders are employed to re-rank both full and partial hypotheses during decoding. Liu *et al.* (2009) proposes a joint-decoding method to combine multiple SMT models into one decoder and integrate translation hypergraphs generated by different models. Both of the last two methods work in a white-box way and need to implement a more complicated decoder to integrate multiple SMT models to work together; meanwhile our method can be conveniently used as a second-pass decoding procedure, without considering any system implementation details.

6 Conclusions and Future Work

In this paper, we have presented a novel MMMBR decoding approach that makes use of a mixture distribution of multiple SMT systems to improve translation accuracy. Compared to single system-based MBR decoding methods, our method can achieve significant improvements on both dev and test sets. What is more, MMMBR decoding approach also outperforms a state-of-the-art system combination method. We have empirically verified that the success of our method comes from both the mixture modeling of translation hypotheses and the combined search space for translation selection.

In the future, we will include more SMT systems with more complicated models into our MMMBR decoder and employ more general MERT algorithms on hypergraphs and lattices (Kumar *et al.*, 2009) for parameter optimization.

References

Chiang David. 2007. *Hierarchical Phrase Based Translation*. *Computational Linguistics*, 33(2): 201-228.

DeNero John, David Chiang, and Kevin Knight. 2009. *Fast Consensus Decoding over Translation Forests*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 567-575.

Hildebrand Almut Silja and Stephan Vogel. 2008. *Combination of Machine Translation Systems via Hypothesis Selection from Combined N-best lists*. In *Proc. of the Association for Machine Translation in the Americas*, pages 254-261.

Huang Liang and David Chiang. 2005. *Better k-best Parsing*. In *Proc. of 7th International Conference on Parsing Technologies*, pages 53-64.

Huang Liang. 2008. *Forest Reranking: Discriminative Parsing with Non-Local Features*. In *Proc. of 46th Meeting of the Association for Computational Linguistics*, pages 586-594.

Koehn Philipp. 2004a. *Phrase-based Model for SMT*. *Computational Linguistics*, 28(1): 114-133.

Koehn Philipp. 2004b. *Statistical Significance Tests for Machine Translation Evaluation*. In *Proc. of Empirical Methods on Natural Language Processing*, pages 388-395.

Kumar Shankar and William Byrne. 2004. *Minimum Bayes-Risk Decoding for Statistical Machine Translation*. In *Proc. of the North American Chapter of the Association for Computational Linguistics*, pages 169-176.

Kumar Shankar, Wolfgang Macherey, Chris Dyer, and Franz Och. 2009. *Efficient Minimum Error Rate Training and Minimum Bayes-Risk Decoding for Translation Hypergraphs and Lattices*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 163-171.

Li Mu, Nan Duan, Dongdong Zhang, Chi-Ho Li, and Ming Zhou. 2009a. *Collaborative Decoding: Partial Hypothesis Re-Ranking Using Translation Consensus between Decoders*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 585-592.

Liu Yang, Haitao Mi, Yang Feng, and Qun Liu. 2009. *Joint Decoding with Multiple Translation Models*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 576-584.

Li Chi-Ho, Xiaodong He, Yupeng Liu, and Ning Xi. 2009b. *Incremental HMM Alignment for MT system Combination*. In *Proc. of 47th Meeting of the Association for Computational Linguistics*, pages 949-957.

Mi Haitao, Liang Huang, and Qun Liu. 2008. *Forest-Based Translation*. In *Proc. of 46th Meeting of the Association for Computational Linguistics*, pages 192-199.

Macherey Wolfgang and Franz Och. 2007. *An Empirical Study on Computing Consensus Translations from multiple Machine Translation Systems*. In *Proc. of Empirical Methods on Natural Language Processing*, pages 986-995.

Och Franz. 2003. *Minimum Error Rate Training in Statistical Machine Translation*. In *Proc. of 41th Meeting of the Association for Computational Linguistics*, pages 160-167.

Och Franz and Hermann Ney. 2004. *The Alignment template approach to Statistical Machine Translation*. *Computational Linguistics*, 30(4): 417-449.

Rosti Antti-Veikko, Spyros Matsoukas, and Richard Schwartz. 2007. *Improved Word-Level System Combination for Machine Translation*. In *Proc. of 45th Meeting of the Association for Computational Linguistics*, pages 312-319.

Roy Tromble, Shankar Kumar, Franz Och, and Wolfgang Macherey. 2008. *Lattice Minimum Bayes-Risk Decoding for Statistical Machine Translation*. In *Proc. of Empirical Methods on Natural Language Processing*, pages 620-629.

Ueffing Nicola, Franz Och, and Hermann Ney. 2002. *Generation of Word Graphs in Statistical Machine Translation*. In *Proc. of Empirical Methods on Natural Language Processing*, pages 156-163.

Wu Dekai. 1997. *Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora*. *Computational Linguistics*, 23(3): 377-404.

Xiong Deyi, Qun Liu, and Shouxun Lin. 2006. *Maximum Entropy based Phrase Reordering Model for Statistical Machine Translation*. In *Proc. of 44th Meeting of the Association for Computational Linguistics*, pages 521-528.

Multi-Sentence Compression: Finding Shortest Paths in Word Graphs

Katja Filippova
Google Inc.
katjaf@google.com

Abstract

We consider the task of summarizing a cluster of related sentences with a short sentence which we call *multi-sentence compression* and present a simple approach based on shortest paths in word graphs. The advantage and the novelty of the proposed method is that it is syntax-lean and requires little more than a tokenizer and a tagger. Despite its simplicity, it is capable of generating grammatical and informative summaries as our experiments with English and Spanish data demonstrate.

1 Introduction

Sentence compression (henceforth SC) is a task where the goal is to produce a summary of a single sentence which would preserve the important part of the content and be grammatical. Starting from the early work of Jing & McKeown (2000), in the last decade SC has received considerable attention in the NLP community. Ubiquitous use of mobile devices is an obvious example of where SC could be applied–a longer text of an email, news or a Wikipedia article can be compressed sentence by sentence to fit into a limited display (Corston-Oliver, 2001). Another reason why SC is so popular is its potential utility for extractive text summarization, single or multi-document (Mani, 2001). There, a standard approach is to rank sentences by importance, cluster them by similarity, and select a sentence from the top ranked clusters. Selected sentences almost always require revision and can be reformulated succinctly as it is often only a part of the sentence which is of interest. It is this multi-document summarization scenario which motivates our work.

Given a cluster of similar, or related, sentences, we aim at summarizing the most salient theme of it in a short single sentence. We refer to this task as *multi-sentence compression*. Defined this way, it comes close to sentence fusion which was originally introduced as a text-to-text generation technique of expressing content common to most of the input sentences in a single sentence (Barzilay & McKeown, 2005). However, since then the technique has been extended so that now fusion also stands for uniting complementary content in a single concise sentence (Filippova & Strube, 2008b; Krahmer et al., 2008). Since our method is not designed for the "union" kind of fusion, we think it is more appropriate to classify it as a sentence compression technique.

Two challenges of SC as well as text summarization are (i) important content selection and (ii) its readable presentation. Most existing systems use syntactic information to generate grammatical compressions. Incidentally, syntax also provides clues to what is likely to be important–e.g., the subject and the verb of the main clause are more likely to be important than a prepositional phrase or a verb from a relative clause. Of course, syntax is not the only way to gauge word or phrase importance. In the case of sentence compression being used for text summarization, one disposes of a rich context to identify important words or phrases. For example, recurring or semantically

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 322–330,
Beijing, August 2010

similar words are likely to be relevant, and this information has been used in earlier SC systems (Hori et al., 2003; Clarke & Lapata, 2007, inter alia). Still, syntactic parsers are assumed to be indispensable tools for both sentence compression and fusion because syntactic constraints (hand-crafted or learned from the data) seem to be the only way to control the grammaticality of the output. In this paper we are going to question this well-established belief and argue that just like in some cases syntax helps to find important content (e.g., when the input is an isolated sentence), in the multi-sentence case redundancy provides a reliable way of generating grammatical sentences. In particular, the important and novel points of our work are as follows:

- We present a simple and robust word graph-based method of generating succinct compressions which requires as little as a part of speech tagger and a list of stopwords.
- To our knowledge, it is the first method which requires neither a parser, nor hand-crafted rules, nor a language model to generate reasonably grammatical output.
- In an extensive evaluation with native speakers we obtain encouraging results for English as well as for Spanish.

In the following section we present our approach to sentence compression (Sec. 2); then we introduce the baseline (Sec. 3) and the data (Sec. 4). In Section 5 we report about our experiments and discuss the results. Finally, Section 6 gives an overview of related work.

2 Multi-sentence Compression

A well-known challenge for extractive multi-document summarization systems is to produce non-redundant summaries. There are two standard ways of avoiding redundancy: either one adds sentences to the summary one-by-one and each time checks whether the sentence is significantly different from what is already there (e.g., using MMR), or one clusters related sentences and selects only one from each cluster. In both cases a selected sentence may include irrelevant information, so one wishes to compress it, usually by taking syntactic and lexical factors into account. However, we think this approach is suboptimal in this case and explore a different way. Instead of compressing a single sentence, we build a *word graph* from all the words of the related sentences and compress this graph.

A word graph is a directed graph where an edge from word *A* to word *B* represents an *adjacency* relation. It also contains the *start* and *end* nodes. Word graphs have been widely used in natural language processing for building language models, paraphrasing, alignment, etc. (see Sec. 6). Compared with dependency graphs, their use for sentence generation has been left largely unexplored, presumably because it seems that almost all the grammatical information is missing from this representation. Indeed, a link between a finite verb and an article does not correspond to any grammatical relation between the two. However, the premise for our work is that redundancy should be sufficient to identify not only important words but also salient links between words. In this section we present our approach to word graph compression. We begin by explaining the graph construction process and continue with the details of two compression methods.

2.1 Word Graph Construction

Given a set of related sentences $S = \{s_1, s_2, ...s_n\}$, we build a word graph by iteratively adding sentences to it. As an illustration, consider the four sentences below and the graph in Figure 1 obtained from them. Edge weights are omitted and italicized fragments from the sentences are replaced with dots for clarity.

(1) *The wife of a former U.S. president Bill* Clinton Hillary Clinton visited China last Monday.

(2) Hillary Clinton wanted to visit China last month *but postponed her plans* till Monday last week.

(3) Hillary Clinton paid *a visit to the People Republic* of China on Monday.

(4) Last week the *Secretary of State* Ms. Clinton visited Chinese officials.

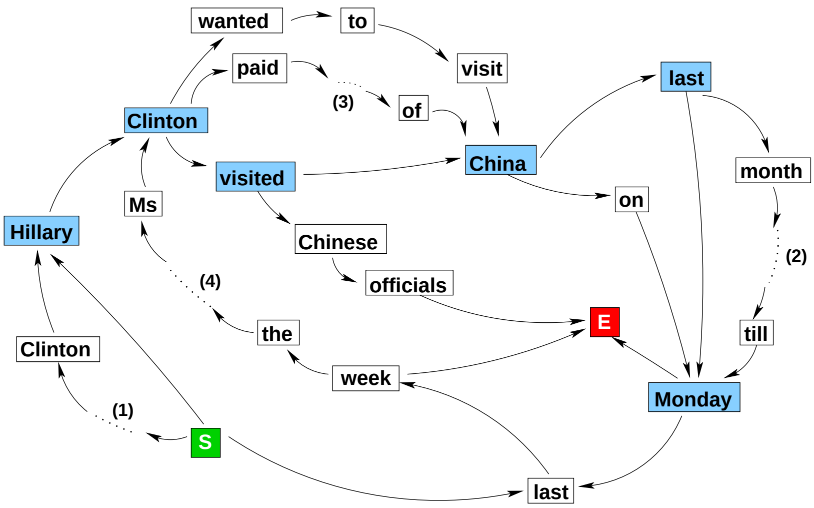


Figure 1: Word graph generated from sentences (1-4) and a possible compression path.

After the first sentence is added the graph is simply a string of word nodes (punctuation is excluded) plus the start and the end symbols (*S* and *E* in Fig. 1). A word from the following sentences is mapped onto a node in the graph provided that they have the exact same lowercased word form and the same part of speech[1] and that no word from this sentence has already been mapped onto this node. Using part of speech information reduces chances of merging verbs with nouns (e.g., *visit*) and generating ungrammatical sequences. If there is no candidate in the graph a new node is created.

Word mapping/creation is done in three steps for the following three groups of words: (1) non-stopwords[2] for which no candidate exists in the graph or for which an unambiguous mapping is possible; (2) non-stopwords for which there are either several possible candidates in the graph or which occur more than once in the sentence; (3) stopwords.

This procedure is similar to the one used by Barzilay & Lee (2003) in that we also first identify "backbone nodes" (unambiguous alignments) and then add mappings for which several possibilities exist. However, they build lattices, i.e., directed acyclic graphs, whereas our graphs may contain cycles. For the last two groups of words where mapping is ambiguous we check the immediate context (the preceding and following words in the sentence and the neighboring nodes in the graph) and select the candidate which has larger overlap in the context, or the one with a greater frequency (i.e., the one which has more words mapped onto it). For example, in Figure 1 when sentence (4) is to be added, there are two candidate nodes for *last*. The one pointing to *week* is selected as *week* is the word following *last* in (4). Stopwords are mapped only if there is some overlap in non-stopword neighbors, otherwise a new node is created.

Once all the words from the sentence are in place, we connect words adjacent in the sentence with directed edges. For newly created nodes, or nodes which were not connected before, we add an edge with a default weight of one. Edge weights between already connected nodes are increased by one. The same is done with the start and end nodes. Nodes store id's of the sentences their words come from as well as all their offset positions in those sentences.

The described alignment method is fairly simple and guarantees the following properties of the word graph: (i) every input sentence corresponds to a loopless path in the graph; (ii) words referring to the same entities or actions are likely to end up in one node; (iii) stopwords are only joined

[1] We use the OpenNLP package for tagging: `http://opennlp.sourceforge.net`.

[2] We generate a list of about 600 news-specific stopwords for English (including, e.g., *said, seems*) and took a publicly available list of about 180 stopwords for Spanish from `www.ranks.nl/stopwords/spanish.html`.

in one node if there is an overlap in context. The graph may generate a potentially endless amount of incomprehensible sequences connecting *start* and *end*. It is also likely to contain paths corresponding to good compressions, like the path connecting the nodes highlighted with blue in Figure 1. In the following we describe two our methods of finding the best path, that is, the best compression for the input sentences.

2.2 Shortest Path as Compression

What properties are characteristic of a good compression? It should neither be too long, nor too short. It should go through the nodes which represent important concepts but should not pass the same node several times. It should correspond to a likely word sequence. To satisfy these constraints we invert edge weights, i.e., link frequencies, and search for the shortest path (i.e., lightest in terms of the edge weights) from *start* to *end* of a predefined minimum length. This path is likely to mention salient words from the input and put together words found next to each other in many sentences. This is the first method we consider. We set a minimum path length (in words) to eight which appeared to be a reasonable threshold on a development set–paths shorter than seven words were often incomplete sentences.

Furthermore, to produce *informative* summaries which report about the main event of the sentence cluster, we filter paths which do not contain a verb node. For example, *Ozark's "Winter's Bone" at the 2010 Sundance Film Festival* might be a good title indicating what the article is about. However, it is not as informative as *"Winter's Bone" earned the grand jury prize at Sundance* which indeed conveys the gist of the event. Thus, we generate *K* shortest paths and filter all those which are shorter than eight words or do not contain a verb. The path with the minimum total weight is selected as the summary.

2.3 Improved Scoring and Reranking

The second configuration of our system employs a more sophisticated weighting function. The purpose of this function is two-fold: (i) to generate a grammatical compression, it favors strong links, i.e., links between words which appear significantly often in this order; (ii) to generate an informative compression, it promotes paths passing through salient nodes.

Strong links: Intuitively, we want the compression path to follow edges between words which are strongly associated with each other. Inverted edge frequency is not sufficient for that because it ignores the overall frequency of the nodes the edge connects. For example, edge frequency of three should count more if the edge connects two nodes with frequency of three rather than if their frequencies are much higher. Thus, we redefine edge weight as follows:

$$w(e_{i,j}) = \frac{\mathtt{freq}(i) + \mathtt{freq}(j)}{\mathtt{freq}(e_{i,j})} \tag{1}$$

Furthermore, we also promote a connection between two nodes if there are multiple paths between them. For example, if some sentences speak of *president Barack Obama* or *president of the US Barack Obama*, and some sentences are about *president Obama*, we want to add some reward to the edge between *president* and *Obama*. However, longer paths between words are weak signals of word association. Therefore, the weight of an edge between the nodes i and j is reduced for every possible path between them but reduced proportionally to its length:

$$w'(e_{i,j}) = \frac{\mathtt{freq}(i) + \mathtt{freq}(j)}{\sum_{s \in S} \mathtt{diff}(s, i, j)^{-1}} \tag{2}$$

where the function $\mathtt{diff}(s, i, j)$ refers to the distance between the offset positions ($\mathtt{pos}(s, i)$) of words i and j in sentence s and is defined as follows:

$$\mathtt{diff}(s, i, j) = \begin{cases} \mathtt{pos}(s, i) - \mathtt{pos}(s, j) & \text{if} \quad \mathtt{pos}(s, i) < \mathtt{pos}(s, j) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Salient words: The function above only indicates how strong the association between two words is. It assigns equal weights to edges connecting words encountered in a single sentence and words encountered next to each other in every sentence. To generate a summary concerning the most salient events and entities, we force the path to go through most frequent nodes by decreasing edge weight with respect to the frequency of the nodes it connects. Thus, we further redefine edge weight as follows:

$$w''(e_{i,j}) = \frac{w'(e_{i,j})}{\texttt{freq}(i) \times \texttt{freq}(j)} \tag{4}$$

We implement the K-shortest paths algorithm to find the fifty shortest paths from *start* to *end* using the weighting function in (4). We filter all the paths which are shorter than eight words and which do not pass a verb node. Finally, we rerank the remaining paths by normalizing the total path weight over its length. This way we obtain the path which has the lightest average edge weight.

3 Baseline

As a first baseline we are searching for the most probable string with respect to the sentence cluster. In particular, we use the Viterbi algorithm to find the sequence of words of a predefined length n which maximizes the bigram probability (MLE-based):

$$p(w_{1,n}) = p(w_1|\mathsf{s})p(w_2|w_1)...p(\mathsf{e}|w_n) \tag{5}$$

Similar to the shortest path implementation, we specify compression length and set it also here to eight tokens. However, the compressions obtained with this method are often unrelated to the main theme. The reason for that is that a token subsequence encountered in a single sentence is likely to get a high probability–all transition probabilities are equal to one–provided that the probability of entering this sequence is not too low. To amend this problem and to promote frequent words (i.e., words which are likely to be related to the main theme) we maximize the following baseline score which takes into account both the bigram probabilities and the token likelihood, $p(w_i)$, which is also estimated from the sentence cluster:

$$b(w_{1,n}) = p(w_1|\mathsf{s})p(w_2|w_1)...p(\mathsf{e}|w_n)\prod_i p(w_i) \tag{6}$$

4 Data Sources

As data for our experiments we use news articles presented in clusters on Google News[3]. The main reason for why we decided to use this service is that it is freely available and does the job of news classification and clustering with a production quality. Apart from that, it is a rich source of multilingual data.

We collected news clusters in English and Spanish, 10-30 articles each, 24 articles on average. To get sets of similar sentences we aggregated first sentences from every article in the cluster, removing duplicates. The article-initial sentence is known to provide a good summary of the article and has become a standard competitive baseline in summarization[4]. Hence, given that first sentences summarize the articles they belong to, which are in turn clustered as concerning the same event, those sentences are likely although not necessarily need to be similar.

From the total of 150 English clusters we reserved 70 for development and 80 for testing. For Spanish we collected 40 clusters, all for testing. We stripped off bylines and dates from the beginning of every sentence with a handful of regular expressions before feeding them to the baseline and our compression methods.

The data we use has two interesting properties: (i) article-initial sentences are on average longer than other sentences. In our case average sentence lengths for English and Spanish (without bylines) are 28 and 35 tokens, respectively. (ii) such sentence clusters are noisier than what one would expect in a summarization pipeline. Both properties make the task realistically hard and pose a challenge for the robustness of a compression method. If we show that reasonable compressions can be generated even from noisy clusters acquired from a publicly available news service, then we have a good reason to believe that the method will perform at least comparable on more carefully constructed clusters of shorter sentences.

[3] `http://news.google.com`

[4] See DUC/TAC competitions: `http://www.nist.gov/tac`

5 Evaluation

5.1 Experiment Design

The performance of the systems was assessed in an experiment with human raters, all native speakers. They were presented with a list of snippets of the articles from one cluster – first sentence and title linked to the original document. The raters were allowed to look up the articles if they need more background on the matter but this was not obligatory.

The first question concerned the quality of the sentence cluster. The raters were asked whether the cluster contained a single prevailing event, or whether it was too noisy and no theme stood out. Given how simple our sentence grouping procedure was, most clusters informed about more than one event. However, to answer the question positively it would be enough to identify one prevailing theme.

Below that, a summary and two further questions concerning its quality were displayed. Similar to most preceding work, we were interested in grammaticality and informativity of summaries. With respect to grammaticality, following Barzilay & McKeown (2005), we asked the raters to give one of the three possible ratings: *perfect* if the summary was a complete grammatical sentence (2 pts); *almost* if it required a minor editing, e.g., one mistake in articles or agreement (1 pt); *ungrammatical* if it was none of above (0 pts). We explicitly asked the raters to ignore lack or excess of capitalization or punctuation. Furthermore, based on the feedback from a preliminary evaluation, we provided an example in which we made clear that summaries consisting of a few phrases which cannot be reformulated as a complete sentence (e.g., *Early Monday a U.S. Navy ship.*) should not count as grammatical.

The final question, concerning informativity, had four possible options: *n/a* if the cluster is too noisy and unsummarizable in the first place; *perfect* if it conveys the gist of the main event and is more or less like the summary the person would produce himself (2 pts); *related* if it is related to the the main theme but misses something important (1 pt); *unrelated* if the summary is not related to the main theme (0 pts).

For each of the 80 sentence clusters (40 for Spanish) we generated three summaries with the three systems. Most summaries were rated by four raters, a few got only three ratings; no rater saw the same cluster twice.

5.2 Results

We report average grammaticality and informativity scores in Table 1. However, averaging system ratings over all clusters and raters is not justified in our case. It is important to remember that the score assignments (i.e., *0, 1, 2*) are arbitrary and that the score of one with respect to grammaticality (i.e., a minor mistake) is in fact closer to two than to zero. One could set the scores differently but even then, strictly speaking, it is not correct to average the scores as ratings do not define a metric space.

System	Gram	Info
Baseline	0.70 / 0.61	0.62 / 0.53
Shortest path	1.30 / 1.27	1.16 / 0.79
Shortest path++	1.44 / 1.25	1.30 / 1.25

Table 1: Average ratings for English / Spanish.

Therefore in Table 2 we present distributions over the three scores for both grammaticality and informativity together with average summary lengths in tokens. For both grammaticality and informativity, for every summary-cluster pair we did majority voting and resolved ties by assigning the lower score. For example, if a system got the ratings *1, 1, 2, 2* for a certain cluster, we counted this as *1*. We dismissed cases where the tie was between the maximum and the minimum score–this happened with some summaries which got just three scores (i.e., *0, 1, 2*) and accounted for $< 4\%$ of the cases. To obtain the informativity distribution we considered only clusters which were classified as containing a single prevailing event by at least ten raters. For English 75 out of 80 clusters qualified as such (37 out of 40 for Spanish). Similar to above, we dismissed about 3% tie cases where the ratings diverged significantly (e.g., *0, 1, 2*).

System	Gram-2	Gram-1	Gram-0	Info-2	Info-1	Info-0	Avg. Len.
Baseline (EN)	21%	15%	65%	18%	10%	73%	8
Shortest path (EN)	52%	16%	32%	36%	33%	31%	10
Shortest path++ (EN)	64%	13%	23%	52%	32%	16%	12
Baseline (ES)	12%	15%	74%	9%	19%	72%	8
Shortest path (ES)	58%	21%	21%	23%	26%	51%	10
Shortest path++ (ES)	50%	21%	29%	40%	40%	20%	12

Table 2: Distribution over possible ratings and average length for English and Spanish.

5.3 Discussion

The difference between the baseline and our shortest path systems is striking. Although more than 20% of the baseline summaries are perfectly grammatical, the gap to the improved version of shortest paths is significant, about 43%. The same holds for the percentage of informative summaries (18% vs. 52%). Both numbers are likely to be understated as we chose to resolve all ties not in our favor. 84% of the summaries generated by the improved method are related to the main theme of the cluster, and more than 60% of those (52% of the total summaries) convey the very gist of it without missing any important information. Comparing the two configurations we have proposed, improved scoring function and reranking we added on top of the shortest path method were both rewarding. Interestingly, even the straightforward approach of choosing the shortest path of a minimum length already guarantees a grammatical summary in more than half of the cases.

An interesting difference in the performance for Spanish and English is that shortest path generates more grammatical sentences than the improved version of it. However, the price for higher grammaticality scores is a huge drop in informativity: half of such summaries are not related to the main theme at all, whereas 40% of the summaries generated by the improved version got the highest rating. A possible reason for the poorer performance for Spanish is that we used a much smaller list of stopwords which did not include news-specific words like, e.g., *dijo* (*said*) which resulted in denser graphs. In the future, we would like to apply the method to more languages and experiment with longer lists of stopwords.

One may notice that the summaries produced by the baseline are shorter than those generated by the shortest paths which might look like a reason for its comparatively poor performance. However, the main source of errors for the baseline was its inability to keep track of the words already present in the summary, so it is unlikely that longer sequences would be of a much higher quality. The sentences generated by the baseline were often repetitive, e.g., *The food tax on food tax on food*. This is not an issue with the shortest path approaches as they never include loops when edge weights are strictly positive.

The reranking we added to the shortest path method is the reason for why the summaries generated by the improved version of the system are on average slightly longer than those produced by the simpler version. The average lengths for both systems are drastically shorter than the average length of the sentences served as input (10/12 vs. 28 tokens in English or 35 tokens for Spanish). This corresponds to the compression rate of 36-43% (29-34% for Spanish) which is comparatively "aggressive" as it usually varies between 50-80% in other systems.

6 Comparison with Related Work

6.1 Sentence Compression

In the last ten years a lot of research has been devoted to sentence compression. Most studies share two properties: (1) they rely on syntax, and (2) they are supervised. The degree of syntax-dependence varies between methods. Some utilize a parser to identify and later keep certain important relations but do not require a complete parse (Clarke & Lapata, 2008), or use a syntactic representation to extract features (McDonald, 2006). For other approaches correct syntactic trees are crucial to obtain grammatical compressions (Galley & McKeown, 2007; Filippova & Strube, 2008a; Cohn & Lapata, 2009). Handcrafted rules (Dorr et al., 2003) as well as language models also have been utilized to generate fluent compressions (Hori et al., 2003; Clarke & Lapata, 2008).

6.2 Sentence Generation

To date the work on sentence fusion is completely dependency syntax-based. Input sentences are parsed into trees, from those trees a new dependency structure is generated, and this structure is finally converted into a sentence (Barzilay & McKeown, 2005; Filippova & Strube, 2008b; Wan et al., 2009). Parser quality is of crucial importance for such methods, and to our knowledge no attempt has been made to generate novel sentences without adhering to dependency representations. In the future, it would be of interest to compare our method with a syntax-based fusion method. Syntax-lean methods have been explored for headline generation (Banko et al., 2000; Dorr et al., 2003; Jin & Hauptmann, 2003). However, they do not aim at generating complete sentences or informative summaries but rather to indicate what the news is about.

6.3 Word Graphs and Lattices

Perhaps the work of Barzilay & Lee (2003) who align comparable sentences to generate sentence-level paraphrases seems closest to ours in that we both use word graphs for text generation. However, this is a fairly general similarity, as both the goal and the implementation are different. While we search for an optimal weighting function in noisy graphs to identify readable and informative compressions, they induce paraphrase patterns from unweighted paths in much smaller DAGs obtained from highly similar sentences. Shen et al. (2006) is another example of using word lattices to find paraphrases. Unlike Barzilay & Lee (2003), they propose to use syntax to obtain accurate alignments. Numerous examples of the utility of word lattices come from the field of finite state automata, language modeling, speech recognition, parsing and machine translation (Mohri, 1997, inter alia).

7 Conclusions

We considered the task of generating a short informative summary for a set of related sentences, called multi-sentence compression, which arises naturally in the context of multi-document text summarization. We presented a simple but robust method which proceeds by finding shortest paths in word graphs. The novelty of our work is that we demonstrated that reasonable compressions can be obtained without any syntactic information if a good weighting function is defined. This distinguishes our work from earlier research on sentence fusion and compression which relies on syntactic representations and/or language models. We provided the details of an extensive evaluation on English and Spanish data and reported high grammaticality as well as informativity scores. In the future we would like to experiment with other languages and eschew using part-of-speech information.

Acknowledgements: I am thankful to Keith Hall for the discussions on this work and the very helpful feedback on an earlier draft of this paper.

References

Banko, M., V. O. Mittal & M. J. Witbrock (2000). Headline generation based on statistical translation. In *Proc. of ACL-00*, pp. 318–325.

Barzilay, R. & L. Lee (2003). Learning to paraphrase: An unsupervized approach using multi-sequence alignment. In *Proc. of HLT-NAACL-03*, pp. 16–23.

Barzilay, R. & K. R. McKeown (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–327.

Clarke, J. & M. Lapata (2007). Modelling compression with discourse constraints. In *Proc. of EMNLP-CoNLL-07*, pp. 1–11.

Clarke, J. & M. Lapata (2008). Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

Cohn, T. & M. Lapata (2009). Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Corston-Oliver, S. H. (2001). Text compaction for display on very small screens. In *Proceedings of the NAACL Workshop on Automatic Summarization*, Pittsburg, PA, 3 June 2001, pp. 89–98.

Dorr, B., D. Zajic & R. Schwartz (2003). Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the Text Summarization Workshop at HLT-NAACL-03*, Edmonton, Alberta, Canada, 2003, pp. 1–8.

Filippova, K. & M. Strube (2008a). Dependency tree based sentence compression. In *Proc. of INLG-08*, pp. 25–32.

Filippova, K. & M. Strube (2008b). Sentence fusion via dependency graph compression. In *Proc. of EMNLP-08*, pp. 177–185.

Galley, M. & K. R. McKeown (2007). Lexicalized Markov grammars for sentence compression. In *Proc. of NAACL-HLT-07*, pp. 180–187.

Hori, C., S. Furui, R. Malkin, H. Yu & A. Waibel (2003). A statistical approach to automatic speech summarization. *EURASIP Journal on Applied Signal Processing*, 2:128–139.

Jin, R. & A. G. Hauptmann (2003). Automatic title generation for spoken broadcast news. In *Proc. of HLT-01*, pp. 1–3.

Jing, H. & K. McKeown (2000). Cut and paste based text summarization. In *Proc. of NAACL-00*, pp. 178–185.

Krahmer, E., E. Marsi & P. van Pelt (2008). Query-based sentence fusion is better defined and leads to more preferred results than generic sentence fusion. In *Proc. of ACL-HLT-08*, pp. 193–196.

Mani, I. (2001). *Automatic Summarization.* Amsterdam, Philadelphia: John Benjamins.

McDonald, R. (2006). Discriminative sentence compression with soft syntactic evidence. In *Proc. of EACL-06*, pp. 297–304.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311.

Shen, S., D. Radev, A. Patel & G. Erkan (2006). Adding syntax to dynamic programming for aligning comparable texts for generation of paraphrases. In *Proc. of COLING-ACL-06*, pp. 747–754.

Wan, S., M. Dras, R. Dale & C. Paris (2009). Improving grammaticality in statistical sentence generation: Introducing a dependency spanning tree algorithm with an argument satisfaction model. In *Proc. of EACL-09*, pp. 852–860.

Enriching Dictionaries with Images from the Internet - Targeting Wikipedia and a Japanese Semantic Lexicon: Lexeed -

Sanae Fujita
NTT Communication Science Lab.
sanae@cslab.kecl.ntt.co.jp

Masaaki Nagata
NTT Communication Science Lab.
nagata.masaaki@lab.ntt.co.jp

Abstract

We propose a simple but effective method for enriching dictionary definitions with images based on image searches. Various query expansion methods using synonyms/hypernyms (or related words) are evaluated. We demonstrate that our method is effective in obtaining high-precision images that complement dictionary entries, even for words with abstract or multiple meanings.

1 Introduction

The Internet is an immense resource for images. If we can form connections between these images and dictionary definitions, we can create rich dictionary resources with multimedia information. Such dictionaries have the potential to provide educational (Popescu et al., 2006), cross-langauge information retrieval (Hayashi et al., 2009) or assistive communication tools especially for children, language learners, speakers of different languages, and people with disabilities such as dyslexia (Mihalcea and Leong, 2008; Goldberg et al., 2009).

Additionally, a database of typical images connected to meanings has the potential to fill the gaps between images and meanings (semantic gap). There are many studies which aim to cross the semantic gap (Ide and Yanai, 2009; Smeulders et al., 2000; Barnard et al., 2003) from the point of view of image recognition. However the semantic classes of target images are limited (e.g. Caltech-101, 256[1]). Yansong and Lapata (2008) tried to construct image databases annotated with keywords from Web news images with their captions and articles, though the semantic coverage is unknown. In this paper, we aim to supply several suitable images for dictionary definitions. We propose a simple but effective method based on an Internet image search.

There have been several studies related to supplying images for a dictionary or thesaurus. Bond et al. (2009) applied images obtained from the Open Clip Art Library (**OCAL**) to Japanese **WordNet**.[2] They obtained candidate images by comparing the hierarchical structures of **OCAL** and **WordNet**, and then judged whether or not the image was suitable for the synset by hand. **OCAL** benefits from being in the public domain; however, it cannot cover a wide variety of meanings because of the limited number of available images.

Fujii and Ishikawa (2005) collected images and text from the Internet by querying lemma, and linked them to an open encyclopedia, **CYCLONE**.[3] They guessed the meaning of the images by disambiguating the surrounding text. This is a straightforward approach, but it is difficult to use it to collect images with minor meanings, because in most cases the Internet search querying lemma only provides images related to the most common meaning. For example, lemma アーチ *arch* may mean `‘‘architecture’’` or `‘‘home run’’` in Japanese, but a lemma search provided no image of the latter at least in the top 500.

There are some resources which link images to target synsets selected from **WordNet** (Fellbaum, 1998). For example, **PicNet** (Borman et al., 2005), **ImageNet** (Deng et al., 2009) and image ontology (Popescu et al., 2006, 2007; Zinger et al., 2006) collect candidate images from the Internet. **PicNet** and **ImageNet** ask Web users to judge their suitability, and Zinger et al. (2006); Popescu et al. (2007) automatically filtered out unsuitable images using visual characteristics. These approaches can

[1] http://www.vision.caltech.edu/Image_Datasets/Caltech101, 256/

[2] http://nlpwww.nict.go.jp/wn-ja/

[3] http://cyclone.cl.cs.titech.ac.jp/

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 331–339,
Beijing, August 2010

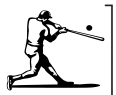
INDEX	アーチ *arch*	(POS: noun)
SENSE 1	DEFINITION	上部$_1$ を 弓$_1$ の 形$_1$ に し た 建物$_1$ 。 また$_9$ 、 その$_3$ 建築$_1$ 様式$_2$ *Buildings with bow-shaped top. Or its architectural style.*
	EXAMPLE	あの$_2$ 橋$_1$ は 2 つ の アーチ$_1$ で 出来$_4$ て いる 。 *That bridge has 2 arches.*
	HYPERNYM	建物$_1$ *building*, 様式$_2$ *style*
	SEM. CLASS	⟨`865:house (main building)`⟩ (⊂ ⟨`2:concrete`⟩), ⟨`2435:pattern, method`⟩ (⊂ ⟨`1000:abstract`⟩)
	IMAGE	
SENSE 3	DEFINITION	野球$_1$ で 、 本塁打$_1$ 。 ホームラン$_1$ 。 *A home run in baseball.*
	EXAMPLE	バッター$_1$ が ライト$_4$ スタンド$_2$ に 逆転$_3$ の アーチ$_3$ を 放っ$_4$ た *A batter blasted the ball over the right-field wall.*
	HYPERNYM	本塁打$_1$ *honruida*
	SYNONYM	ホームラン$_1$ *home run*, DOMAIN 野球$_1$ *baseball*
	SEM. CLASS	⟨`1680:sport`⟩ (⊂ ⟨`1000:abstract`⟩)
	IMAGE	

Figure 1: Simplified Entry for **Lexeed** & **Hinoki**:アーチ *arch*

collect a large number of highly accurate images. However, target synsets are limited at present, and the coverage of polysemous words is unknown. We present a comparison with **ImageNet** and image ontology (Popescu et al., 2006) in § 3.

In this paper, to cover a broad range of meanings, we use an Internet search. In advance, we expand the number of queries per meaning using information extracted from definition sentences. In § 3, we investigate the usability and effectiveness of several types of information targeting two different types of dictionaries, a Japanese Semantic Lexicon: **Lexeed** and a Web Dictionary: Japanese **Wikipedia**[4] (§ 2). We show that our method is simple but effective. We also analyze senses that are difficult to portray using images.

2 Resources

2.1 Japanese Semantic Lexicon: Lexeed

We use **Lexeed**, a Japanese Semantic Lexicon (Kasahara et al., 2004) as a target dictionary (see Figure 1). **Lexeed** includes the 29,000 most familiar words in Japanese, split into 48,000 senses. Each entry contains the word itself and its part of speech (POS) along with definition and example sentences and links to the Goi-Taikei (**GT**) Japanese Ontology (Ikehara et al., 1997). In addition, we extracted related words such as hypernyms, synonyms, and domains, from the definitions (called **Hinoki** Ontology). The images in Figure 1 are samples provided using our method.

[4]http://ja.wikipedia.org/

Table 1: Size of **Lexeed** and Japanese **Wikipedia** (disambiguation)

No.	**Lexeed**	**Wikipedia**	Shared Lemma
Entries	29,272	33,299	2,228
Senses	48,009	197,912[1]	19,703
Ave. Senses/Entry	1.6	5.9	8.8
Max. Senses/Entry	57	320	148
Monosemous	19,080	74	2
Ave. Words/Definition[2]	14.4	10.7	11.0

[1]From the all 215,883 lists, we extracted lists showing senses obtained by heuristics (see lines 2,3,4,6,7,9 and 10 for Figure 2).
[2]Analyzed by Mecab, http://mecab.sourceforge.net/

2.2 Web Dictionary :Japanese Wikipedia

We used **Wikipedia**'s disambiguation pages,[5] as a target dictionary (see Figure 2). A disambiguation page lists articles (eg. `''European Union''`, `''Ehime University''`) associated with the same lemma (eg. "EU"). Our goal is to provide images for each article listed. As shown in Figure 2, they include various writing styles.

2.3 Comparison of Lexeed and Wikipedia

Table 1 shows the sizes of **Lexeed** and **Wikipedia**'s disambiguation pages, and the shared entries. Shared entries are rare, and account for less than

[5]Version 20091011.

Original (in Japanese)

```
1  '''EU'''
2  * [[欧州連合]]
3  * [[Europa Universalis]]シリーズ - [[パラ
   ドックスインタラクティブ]]の[[歴史シミュレー
   ションゲーム]]
4  * [[愛媛大学]](Ehime University) - [[愛媛
   県]][[松山市]]にある日本の[[国立大学]]
5  '''Eu'''
6  * [[ユウロピウム]]の元素記号
7  * [[ユーフォニアム]] - 金管楽器
8 '''eu'''
9 * [[.eu]] - 欧州連合の[[国別ドメイン]]
10 * [[バスク語]]の[[ISO 639|ISO 639-1
   言語コード]]
```

Gloss

```
1 '''EU'''
2 * [[European Union]]
3 * [[Europa Universalis]] series - a [[histori-
  cal computer game]] by [[Paradox Interactive]]
4 * [[Ehime University]] - a [[National Univer-
  sity]] in [[Matsuyama]],[[Ehime Prefecture]]
5 '''Eu'''
6 * [[Europium]]'s chemical element symbol
7 * [[euphonium]] - a brass instrument
8 '''eu'''
9 * [[.eu]] - [[country-code top-level domain]]
  for the European Union
10 * [[ISO 639|ISO 639-1 language code]] of
  [[Basque]]
```

[[ ]] shows a link in **Wikipedia**. And we assign each line a number for easy citation.

Figure 2: Simplified Example of **Wikipedia**'s Disambiguation Page: "EU (disambiguation)"

10 % of the total [6][7]. As regards **Lexeed**, 16,685 entries (57 %) do not appear in any of **Wikipedia**'s lemmas, not only in disambiguation pages.[8]

As shown in Table 1, **Wikipedia** has many senses, but most of them are proper nouns. For example, in **Lexeed**, ヒマワリ *sunflower* is monosemous, but in **Wikipedia**, 67 senses are listed, including 65 proper nouns besides ``plant'' and ``sunflower oil''. On the other hand, in **Wikipedia**, アーチ *arch* has only one sense, ``architecture'' corresponding to **Lexeed**'s アーチ$_1$ *arch*, and has no disambiguation page.

As mentioned above, **Lexeed** and **Wikipedia** have very different types of entries and senses. This research aims to investigate the possibility of supplying appropriate images for such different senses, and a method for obtaining better images.

3 Experiment to Supply Images for Word Senses

In this paper, we propose a simple method for supplying appropriate images for each dictionary sense of a word. We collect candidate images from the Internet by using a querying image search. To obtain images even for minor senses, we expand the query by appending queries extracted from definitions for each sense.

In this paper, we investigated two main types of expansion, that is, the appending of mainly synonyms (SYN), and related words including hypernyms (LNK). For information retrieval, query expansion using synonyms has been adopted in several studies (Voorhees, 1994; Fang and Zhai, 2006; Unno et al., 2008). Our LNK is similar to methods used in Deng et al. (2009), but we note that their goal is not to give images to polysemous words (which is our intention). Popescu et al. (2006) also used synonyms (all terms in a synset) and hypernyms (immediate supertype in **WordNet**), but they did not investigate the effectiveness of each expansion and they forcus only on selected object synsets.

3.1 Experimental and Evaluation Method

We collected five candidate images for each sense from the Internet by querying an image search engine.[9] Then we manually evaluated the suitability of the image for explaining the target sense. The evaluator determined whether or not the image was appropriate (**T**), acceptable (**M**), or inappropriate (**F**). The evaluator also noted the reasons for **F**.

Figure 3 shows an example for たまねぎ *onion*. As shown in Figure 3, the evaluator determined **T**, **M** or **F** for each candidate image.

[6]Shared lemmas are そば *buckwheat noodle*, サイクル *cycle*, フクロウ *owl*, etc.

[7]Lemmas only in **Wikipedia** are イソップ *Aesop*, ビオ *Biot/Veoh*, 竜門の滝 *fall name*, etc.

[8]Lemmas only in **Lexeed** are 後払い *pay later*, ユーモラス *humorous*, 抜擢 *selection*, etc.

[9]We used Google AJAX images API, http://code.google.com/intl/ja/apis/ajaxsearch/

(1) **T** (Appropriate) (2) **F** (Inappropriate) (3) **M** (Acceptable) (4) **T** (Appropriate) (5) **T** (Appropriate)

Figure 3: Examples of Candidate Images and Evaluations for たまねぎ *onion*

Table 2: Data for **Hinoki** Ontology

Type	No.	%	Example: Lemma	Example: Related Word
Hypernym	47,054	69.1	アーチ$_1$ *arch*	様式
Synonym	14,068	20.6	アーチ$_3$ *arch*	ホームラン *homer*
Domain	1,868	2.7	アーチ$_3$ *arch*	野球 *baseball*
Hyponym	757	1.1	売り買い$_1$ *buy and sell*	売る *sell*
Meronym	686	1.0	赤身$_1$ *lean*	魚肉 *fish meat*
Abbreviation	383	0.6	亜$_2$ *A(sia)*	アジア *Asia*
Other name	216	0.3	差し込み$_2$ *shave*	コンセント *plug outlet*
Other	3102	4.6	包み焼き$_1$ *papillote*	魚 *fish*
Total	68,134	100		

For an image that is related but that does not explain the sense, the evaluation is **F**. For example, for たまねぎ *onion*, the images of onion dishes such as (2) in Figure 3 are **F**. On the other hand, the images that show onions themselves such as (1), (4) and (5) in Figure 3 are **T**. With (3) in Figure 3, the image may show the onion itself or a field of onions, therefore the evaluation is **M**.

One point of judgment, specifically between **T** and **M**, is whether the image is typical or not. With たまねぎ *onion*, most typical images are similar to (1), (4) and (5). The image (3) may not be typical but is helpful for understanding, and (2) may lead to a misunderstanding if this is the only image shown to the dictionary user. This is why (3) is judged to be **M** and (2) is judged to be **F**.

We evaluated 200 target senses for **Lexeed**, and 100 for **Wikipedia**.[10]

3.2 Experiment: Lexeed

In this paper, we expand queries using the **Hinoki** Ontology (Bond et al., 2004), which includes related words extracted from the definition sentences. Table 2 shows the data for the **Hinoki** Ontology.

For SYN, we expand queries using synonyms, abbreviations, other names in Table 2, and variant spellings found in the dictionary. On the other hand, for LNK, we use all the remaining relations, namely hypernyms, domains, etc. Additionally, we use only normal spellings with no expansion, when the target words are monosemous (MONO). One exception should be noted. When the normal spelling employs hiragana (Japanese syllabary characters), we expand it using a variant spelling. For example, とんぼ *dragonfly* is expanded by the variant spelling 蜻蛉 *dragonfly*.

To investigate the trends and difficulties based on various conditions, we split the **Lexeed** senses into four types, namely, concrete and monosemous (MC), or polysemous (PC), not concrete and monosemous (MA), or polysemous (PA). We selected 50 target senses for evaluation randomly for each type. The target senses were randomly selected without distinguishing them in terms of their POS.

Note that we regard the sense as being something concrete that is linked to **GT**'s semantic classes subsumed by ⟨`2:concrete`⟩, such as たまねぎ *onion* (⊂ ⟨`677:crop/harvest/farm products`⟩ ⊂ ⟨`2:concrete`⟩).

3.3 Results and Discussion: Lexeed

Table 3 shows the ratio of **T** (appropriate), **M** (acceptable) and **F** (inappropriate) images for the target sense. We calculated the ratio using all five candidate images, for example, in Figure 3, the

[10]We performed an image search in September 2009 for **Lexeed**, and in December 2009 for **Wikipedia**.

ratio of appropriate images is 60 % (three of five).

In Table 3, the baseline shows a case where the query only involves the lemma (normal spelling). As shown in Table 3, SYN has higher precision than LNK. This means that SYN can focus on the appropriate sense. With polysemous words (PC, PA), expansion works more effectively, and helps to supply appropriate images for each sense. However, with MC, both LNK and SYN have less precision. This is because the target senses of MC are majorities, so expansion is adversely affected. Although MONO alone has good precision, because hiragana is often used as readings and has high ambiguity, appending the variant spelling helps us to focus on the appropriate sense.

Here, we focus on LNK of PC, and then analyze the reasons for **F** (Table 5). In Table 5, in 24.3% of cases it is "difficult to portray the sense using images" (The numbers of senses for which it is "difficult to portray the sense using images" are, 3 of MC, 9 of PC, 10 of MA, and 16 of PA. We investigate such senses in more detail in § 3.4.).

For such senses, no method can provide suitable images, as might be expected. Therefore, we exclude targets where it is "difficult to portray the sense using images", then we recalculated the ratio of appropriate images. Table 4 shows the capability of our proposed method for senses that can be explored using images. This leads to 66.3 % precision (15.3% improvement) even for most difficult target type, PA.

Again, when we look at Table 5, reasons 2-5 (33.3 %) will be improved. In particular, "hypernym leads to ambiguity" makes up more than 10%. Hypernyms sometimes work well, but sometimes they lead to other words included in the hypernyms. For example, appending the hypernym 食品 *foods* to 煮干し *boiled-dried fish* leads to images of "foods made with boiled-dried fish". This is why SYN obtained better results than LNK. Then, with "expanded by minor sense" and when the original sense is dominant majority, expansion reduced the precision. Therefore, we should expand using only words with major senses.

3.4 Discussion: Senses can/cannot be shown by images

As described above, the target senses are randomly selected without being distinguished by their POS, because we also want to investigate the features of senses that can be shown by images. Table 6 shows the ratio of senses judged as "difficult to portray the sense using images" (labeled as "Not Shown") for each POS. As regards POS, the majority of selected senses are nouns, followed by verbal nouns and verbs. We expected that the majority of nouns and verbal nouns whould be "Shown", but did not expect that a majority of verb is also "Shown". Other POSs are too rare to judge, although they tend to fall in the "Not Shown" category.

Furthermore, in Table 7, for nouns and verbal nouns, we show the ratio of senses for each type ("Concrete" or "not Concrete") judged in terms of "difficult to portray the sense using images". We classified the senses into "Concrete" or "not Concrete" based on **GT**'s semantic classes, as described in § 3.2.

Table 6: Ratio of Senses judged as "difficult to portray the sense using images" for each POS

POS	Shown		Not Shown		Total
	No.	%	No.	%	No.
Noun	132	85.2	23	14.8	155
Verbal Noun	15	78.9	4	21.1	19
Verb	9	81.8	2	18.2	11
Affix	4	57.1	3	42.9	7
Pronoun	0	0	2	100	2
Adjective	1	50	1	50	2
Adverb	0	0	2	100	2
Interjection	1	100	0	0	1
Conjunction	0	0	1	100	1
Total	162	81	38	19	200

Table 7: Ratio of Concrete/Not Concrete Senses judged as "difficult to portray the sense using images": for Nouns and Verbal Nouns

Type	Shown		Not Shown		Total
	No.	%	No.	%	No.
Concrete	114	90.5	12	9.5	126
Not Concrete	33	68.8	15	31.3	48
Total	147	84.5	27	15.5	174

Table 3: Ratio of Appropriate Images for Sense (Precision): **Lexeed**

Target Type		Expanding Method	**F** (Inappropriate) No.	%	**T** (Appropriate) No.	%	**M** (Acceptable) No.	%	**T+M** No.	%	Total
Con-crete	Mono-semous (MC)	SYN	18	24.0	36	48.0	21	28.0	57	76.0	75
		LNK	82	**33.5**	112	45.7	51	20.8	163	66.5	245
		MONO	42	16.8	181	**72.4**	27	10.8	208	**83.2**	250
		baseline	46	18.4	171	68.4	33	13.2	204	81.6	250
	Poly-semous (PC)	SYN	94	38.7	88	36.2	61	25.1	149	**61.3**	243
		LNK	111	**44.4**	92	36.8	47	18.8	139	55.6	250
		baseline	180	**72.0**	53	21.2	17	6.8	70	28.0	250
not Con-crete	Mono-semous (MA)	SYN	32	42.7	21	28.0	22	29.3	43	**57.3**	75
		LNK	138	**57.5**	54	22.5	48	20.0	102	42.5	240
		MONO	98	40.0	98	40.0	49	20.0	147	**60.0**	245
		baseline	112	44.8	86	34.4	52	20.8	138	55.2	250
	Poly-semous (PA)	SYN	122	49.0	64	25.7	63	25.3	127	**51.0**	249
		LNK	150	**60.2**	52	20.9	47	18.9	99	39.8	249
		baseline	201	**80.7**	36	14.5	12	4.8	48	19.3	249

Table 4: Ratio of Appropriate Images for Sense (Precision), excluding senses that are difficult to portray using images: **Lexeed**

Target Type		Expanding Method	**F** (Inappropriate) No.	%	**T** (Appropriate) No.	%	**M** (Acceptable) No.	%	**T+M** No.	%	Total
Con-crete	Mono-semous (MC)	SYN	15	**21.4**	36	51.4	19	27.1	55	**78.6**	70
		LNK	71	**30.9**	112	48.7	47	20.4	159	69.1	230
		MONO	29	12.3	180	76.6	26	11.1	206	**87.7**	235
		baseline	35	14.9	170	72.3	30	12.8	200	85.1	235
	Poly-semous (PC)	SYN	61	30.8	85	42.9	52	26.3	137	**69.2**	198
		LNK	84	40.0	89	42.4	37	17.6	126	60.0	210
		baseline	139	67.8	53	25.9	13	6.3	66	32.2	205
not Con-crete	Mono-semous (MA)	SYN	17	34.0	20	40.0	13	26.0	33	**66.0**	50
		LNK	101	**51.8**	54	27.7	40	20.5	94	48.2	195
		MONO	65	33.3	94	48.2	36	18.5	130	**66.7**	195
		baseline	72	36	85	42.5	43	21.5	128	64.0	809
	Poly-semous (PA)	SYN	57	33.7	63	37.3	49	29	112	**66.3**	169
		LNK	81	47.9	52	30.8	36	21.3	88	52.1	169
		baseline	122	72.2	36	21.3	11	6.5	47	27.8	169

Table 5: Reasons for **F**: PC, LNK:**Lexeed**

No.	Reason	No.	%	Example
1	difficult to portray the sense using images	27	24.3	これ *me* ``humble expressions used for oneself''
2	hypernym leads to ambiguity	12	10.8	煮干し *boiled-dried fish* (⊂ 食品 *foods*)
3	expanded by minor sense	11	9.9	リンク *link* (⊂ リンクス *links*, usually means *lynx*)
4	no expansion is better	8	7.2	カメラマン *cameraman* (⊂ 部員 *staff*)
5	original sense is TOO minor	6	5.4	海 *lake* (⊂ 湖 *lake*), 海 usually means *sea*
6	Other	47	42.3	
	Total	111	100	

As shown in Table 7, 90.5 % of "Concrete" nouns are judged as "Shown", and only 9.5 % of senses are judged as "Not Shown" [11]. However 68.8 % of "not Concrete" nouns are also judged as "Shown".

Therefore, both POS and type ("Concrete" or "not Concrete") are helpful, but not perfect features as regards knowing the sense is "difficult to portray the sense using images". In future work we will undertake further analysis to determine the critical features.

3.5 Experiment: Wikipedia

For LNK we use the **Wikipedia** hyperlinks (shown as [[ ]] in Fig 2). 95.5 % of all senses include [[ ]], 85.4 % linked to an actual page, and [[ ]] appeared 0.95 times per sense. Note that we do not use time expression links such as [[2010]] and [[1990s]].

With SYN, we use synonyms extracted with heuristics. Table 8 shows the main rules that we used to extract synonyms. We extracted synonyms for 98.0 % of 197,912 senses.

Then we randomly selected 50 target senses for evaluation from lemmas shared/unshared by **Lexeed**.

3.6 Results and Discussion: Wikipedia

We do not show the baseline in Table 9, but it is always below 10%. For all target senses, expansion provides more suitable images. Because there are so many senses in **Wikipedia**, no target sense is in the majority. As shown in Table 9, there are few differences between SYN and LNK, because most of the synonyms used for SYN are also links. However, SYN has slightly superior precision as regards **T** (Appropriate), which means the process of extracting synonyms helped to reject links that were poorly with the target senses.

Also in **Lexeed**, expansion using synonyms (SYN) had higher precision than hypernyms (LNK). Because we do not know the total number of suitable images for the target senses on the Internet, we cannot estimate the recall with this evaluation method. However, we speculate that hypernyms provide higher recall. Deng et al. (2009) undertook expansion using hypernyms and this may be an appropriate way to obtain many more images for each sense. However, because our aim is employ several suitable images for each sense, high precision is preferable to high recall.

Now, we focus on LNK shared by **Lexeed**, and then we analyze the reasons for **F** (Table 10). In contrast to **Lexeed**, no sense is classified as "difficult to portray the sense using images". However, there are many senses where it is difficult to decide what kind of images "explain the target sense". For example, in Table 10, with "maybe **T** (Appropriate)", the target sense was a personal name and the image was his/her representative work. In this paper, for personal names, only the images of the person are judged to be **T**, despite the fact that supplying images of representative work for novelists or artists may be suitable.

In this study, we obtained five images per sense, but only one image was sufficient for some senses, for example, an image of an album cover for the name of an album. In contrast, several different types of images are needed for some senses. For example, for the name of a city, images of maps, landscapes, city offices, symbols of the city, etc. are all suitable. Therefore, it may be better to estimate a rough class first, such as the name of an album, artist and place, and then obtain preassigned types of images.

4 Conclusions

The goal of this work was to supply several suitable images for dictionary definitions. The target dictionaries were **Lexeed** and **Wikipedia**, which have very different characteristics. To cover a wide range of senses, we collected candidate images from the Internet by querying an image search engine. Then, to obtain suitable and different images for each sense, we expanded the queries by appending related words extracted from the definition sentences. In this paper, we tried two types of expansion, one mainly using synonyms (SYN), and one mainly using hypernyms or related links (LNK).

The results show that SYN provided better precision than LNK, especially for **Lexeed**. Also, query expansion provided a substantial improvement for

[11] For example, 学会 *conference* (⊂ ⟨`373:organization, etc.`⟩ ⊂ ⟨`2:concrete`⟩), 親代わり *parental surrogate* (⊂ ⟨`342:agent/representative`⟩ ⊂ ⟨`2:concrete`⟩), and so on.

Table 8: Rules for Extracting Synonyms for SYN: **Wikipedia**

Rule	Example: Lemma	Example: Definition sentences
head parts separated by hyphen (- or –)	EU	[[euphonium]] - a brass instrument (line 7 in Figure 2)
whole definitions appear as a chunk	EU	[[European Union]] (line 2 in Figure 2)
parts indicated by		
arrow (→)	イヌ *dog*	One of [[Oriental Zodiac]]→[[戌 *dog*]]
quotation key words, 参照 *See* etc.	イヌ *dog*	[[Chinese character]]'s [[radical parts]], See [[犬部 *inu-bu*]]
parts in parentheses or " " including		
whole lemma	Einstein	"Albert Einstein"
alphameric characters, for katakana lemma	サンバ	"samba"
characters of alpha-numeral lemma	CS	コンピュータ科学 (computer science)

underlined parts show the extracted synonyms.

Table 9: Ratio of Appropriate Images for Sense (Precision): **Wikipedia**

Target Type	Expanding Method	**F** (Inappropriate) No.	%	**T** (Appropriate) No.	%	**M** (Acceptable) No.	%	**T+M** No.	%	Total
Shared by	SYN	98	40.8	119	**49.6**	23	9.6	142	**59.2**	240
Lexeed	LNK	92	41.8	107	48.6	21	9.5	128	58.2	220
NOT shared	SYN	100	41.2	103	**42.4**	40	16.5	143	58.8	243
by **Lexeed**	LNK	96	41.0	93	39.7	45	19.2	138	**59.0**	234

Table 10: Reasons for **F**: Shared by **Lexeed**, LNK: **Wikipedia**

No.	Reason	No.	%	Example: Lemma	Example: Links
7	lack of queries (available words in def.)	14	15.2	ふえ *fue (reading)*	フエ *Hue, city name in Vietnam*
8	inappropriate queries (available words in def.)	10	10.9	レギュラー *regular*	出場選手登録 *active roster*
2	hypernym lead to ambiguity	5	5.4	キャッシュ *cache*	ジオキャッシング *geocaching*
9	maybe **T** (Appropriate)	5	5.4	モンキー *monkey*	モンキー・パンチ *Monkey Punch*
6	Other	58	63		
	Total	92	100		

polysemous words. Our proposed method is simple but effective for our purpose, that is supplying suitable and different images for each sense.

In future work we intend to analyze senses that are difficult/easy to portray using images in more detail, using not only semantic charactaristics but also visual features(Csurka et al., 2004). We also intend to improve the expansion method. One way to achieve this is to filter out expansions with minor senses. As for **Wikipedia**, we should approximate the class first, such as the name of an album, artist and place, then obtain preassigned types of images.

References

Kobus Barnard, Pinar Duygulu, Nando de Freitas, David Forsyth, David Blei, and Michael I. Jordan. 2003. Matching Words and Pictures. *Journal of Machine Learning Research*, Vol. 3, pp. 1107–1135.

Francis Bond, Hitoshi Isahara, Sanae Fujita, Kiyotaka Uchimoto, Takayuki Kuribayashi, and Kyoko Kanzaki. 2009. Enhancing the Japanese WordNet. In *The 7th Workshop on Asian Language Resources, in conjunction with ACL-IJCNLP-2009*, pp. 1–8.

Francis Bond, Eric Nichols, Sanae Fujita, and Takaaki Tanaka. 2004. Acquiring an Ontology for a Fundamental Vocabulary. In *Proceedings of the 20th International Conference on Computational Linguistics: COLING-2004*, pp. 1319–1325.

Andy Borman, Rada Mihalcea, and Paul Tarau. 2005. PicNet: Pictorial Representations for Illustrated Semantic Networks. In *Proceedings of the AAAI Spring Symposium on Knowledge Collection from Volunteer Contributors*.

Gabriela Csurka, Cedric Bray, Chris Dance, and Lixin Fan. 2004. Visual categorization with bags of keypoints. In *ECCV International Workshop on Statistical Learning in Computer Vision*, pp. 59–74.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Computer Vision and Pattern Recognition (CVPR)*.

Hui Fang and ChengXiang Zhai. 2006. Semantic term matching in axiomatic approaches to information retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pp. 115–122. ACM.

Christine Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Atsushi Fujii and Tetsuya Ishikawa. 2005. Image Retrieval and Disambiguation for Encyclopedic Web Search. In *Proceedings of the International Joint Conference on Artificial Intelligence: IJCAI-2005*, pp. 1598–1599.

Andrew B. Goldberg, Jake Rosin, Xiaojin Zhu, and Charles R. Dyer. 2009. Toward Text-to-Picture Synthesis. In *NIPS 2009 Mini-Symposia on Assistive Machine Learning for People with Disabilities*.

Yoshihiko Hayashi, Savas Bora, and Masaaki Nagata. 2009. Utilizing Images for Assisting Cross-language Information Retrieval on the Web. In *International Workshop on Web Information Retrieval Support Systems*, pp. 100–103.

Ichiro Ide and Keiji Yanai. 2009. Crossing the Semantic Gap : Towards the Understanding of Image and Video Contents. *Journal of Japanese Society for Artificial Intelligence*, Vol. 24, No. 5, pp. 691–699. (in Japanese).

Satoru Ikehara, Masahiro Miyazaki, Satoshi Shirai, Akio Yokoo, Hiromi Nakaiwa, Kentaro Ogura, Yoshifumi Ooyama, and Yoshihiko Hayashi. 1997. *Goi-Taikei — A Japanese Lexicon*. Iwanami Shoten, Tokyo. 5 volumes/CD-ROM.

Kaname Kasahara, Hiroshi Sato, Francis Bond, Takaaki Tanaka, Sanae Fujita, Tomoko Kanasugi, and Shigeaki Amano. 2004. Construction of a Japanese Semantic Lexicon: Lexeed. In *IEICE Technical Report: 2004-NLC-159*, pp. 75–82. (in Japanese).

Rada Mihalcea and Chee Wee Leong. 2008. Toward communicating simple sentences using pictorial representations. *Machine Translation*, Vol. 22, No. 3, pp. 153–173.

Adrian Popescu, Christophe Millet, and Pierre-Alain Moëllic. 2007. Ontology Driven Content Based Image Retrieval. In *Proceedings of the ACM International Conference on Image and Video Retrieval*.

Adrian Popescu, Christophe Millet, Pierre-Alain Moëllic, Patrick Hède, and Gregory Grefenstette. 2006. Automatic Construction of a Grounded Multimedia Ontology of Objects to Illustrate Concepts in a Learning Process. In *NETTIES 2006 Conference: Advanced Educational Technologies for a Future e-Europe*.

Arnold W.M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. 2000. Content-based Image Retrieval at the End of the Early Years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 12, pp. 1349–1380.

Yuya Unno, Yusuke Miyao, and Jyunichi Tujii. 2008. Information Retrieval using Automatically Extracted paraphrases. In *Proceedings of the 14th Annual Meeting of The Association for Natural Language Processing: NLP-2008*, pp. 123–126. (in Japanese).

Ellen M. Voorhees. 1994. Query Expansion using Lexical-Semantic Relations. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, pp. 61–69.

Feng Yansong and Mirella Lapata. 2008. Automatic image annotation using auxiliary text information. In *Proceedings of ACL-08: HLT*, pp. 272–280. Association for Computational Linguistics.

Svitlana Zinger, Christophe Millet, Benoit Mathieu, Gregory Grefenstette, Patrick Hède, and Pierre-Alain Moëllic. 2006. Clustering and semantically filtering web images to create a large-scale image ontology. In *SPIE 18th Annual Symposium Electronic Imaging, Internet Imaging VII*, Vol. 6061, pp. 89–97.

Opinosis: A Graph-Based Approach to Abstractive Summarization of Highly Redundant Opinions

Kavita Ganesan and **ChengXiang Zhai** and **Jiawei Han**
Department of Computer Science
University of Illinois at Urbana-Champaign
{kganes2,czhai,hanj}@cs.uiuc.edu

Abstract

We present a novel graph-based summarization framework (Opinosis) that generates concise abstractive summaries of highly redundant opinions. Evaluation results on summarizing user reviews show that Opinosis summaries have better agreement with human summaries compared to the baseline extractive method. The summaries are readable, reasonably well-formed and are informative enough to convey the major opinions.

1 Introduction

Summarization is critically needed to help users better digest the large amounts of opinions expressed on the web. Most existing work in Opinion Summarization focus on predicting sentiment orientation on an entity (Pang et al., 2002) (Pang and Lee, 2004) or attempt to generate aspect-based ratings for that entity (Snyder and Barzilay, 2007) (Lu et al., 2009)(Lerman et al., 2009)(Titov and Mcdonald, 2008). Such summaries are very informative, but it is still hard for a user to understand why an aspect received a particular rating, forcing a user to read many, often highly redundant sentences about each aspect. To help users further digest the opinions in each aspect, it is thus desirable to generate a concise textual summary of such redundant opinions.

Indeed, in many scenarios, we will face the problem of summarizing a large number of highly redundant opinions; other examples include summarizing the 'tweets' on Twitter or comments made about a blog or news article. Due to the subtle variations of redundant opinions, typical extractive methods are often inadequate for summarizing such opinions. Consider the following sentences:

1. *The iPhone's battery lasts long, only had to charge it once every few days.*
2. *iPhone's battery is bulky but it is cheap.*.
3. *iPhone's battery is bulky but it lasts long!*

With extractive summarization, no matter which single sentence of the three is chosen as a summary, the generated summary would be biased. In such a case, an abstractive summary such as '*iPhone's battery is cheap, lasts long but is bulky*' is a more complete summary, conveying all the necessary information. Extractive methods also tend to be verbose and this is especially problematic when the summaries need to be viewed on smaller screens like on a PDA. Thus, an informative and concise abstractive summary would be a better solution.

Unfortunately, abstractive summarization is known to be difficult. Existing work in abstractive summarization has been quite limited and can be categorized into two categories: (1) approaches using prior knowledge (Radev and McKeown, 1998) (Finley and Harabagiu, 2002) (DeJong, 1982) and (2) approaches using Natural Language Generation (NLG) systems (Saggion and Lapalme, 2002) (Jing and McKeown, 2000). The first line of work requires considerable amount of manual effort to define schemas such as frames and templates that can be filled with the use of information extraction techniques. These systems were mainly used to summarize news articles. The second category of work uses deeper NLP analysis with special techniques for text regeneration. Both approaches either heavily rely on manual effort or are domain dependent.

In this paper, we propose a novel flexible summarization framework, Opinosis, that uses graphs to produce abstractive summaries of highly redundant opinions. In contrast with the previous work, Opinosis assumes no domain knowledge and uses shallow NLP, leveraging mostly the word order in the existing text and its inherent redundancies to generate informative abstractive summaries. The key idea of Opinosis is to first construct a textual graph that represents the text to be summarized. Then, three unique properties of this graph are used to explore and score various subpaths that help in generating candidate abstractive summaries.

Evaluation results on a set of user reviews show that Opinosis summaries have reasonable agreement with human summaries. Also, the gener-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 340–348,
Beijing, August 2010

ated summaries are readable, concise and fairly well-formed. Since Opinosis assumes no domain knowledge and is highly flexible, it can be potentially used to summarize any highly redundant content and could even be ported to other languages. (All materials related to this work including the dataset and demo software can be found at `http://timan.cs.uiuc.edu/downloads.html`.)

2 Opinosis-Graph

Our key idea is to use a graph data structure (called Opinosis-Graph) to represent natural language text and cast this abstractive summarization problem as one of finding appropriate paths in the graph. Graphs have been commonly used for extractive summarization (e.g., LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004)), but in these works the graph is often *undirected* with *sentences as nodes* and similarity as edges. Our graph data structure is different in that each node represents a *word unit* with *directed edges* representing the structure of sentences. Moreover, we also attach positional information to nodes as will be discussed later.

Algorithm 1 (A1): $OpinosisGraph(Z)$

1: **Input:** Topic related sentences to be summarized: $Z = \{z_i\}_{i=1}^{n}$
2: **Output:** $G = (V, E)$
3: **for** $i = 1$ to n **do**
4: $\quad w \leftarrow Tokenize(z_i)$
5: $\quad sent\_size \leftarrow SizeOf(w)$
6: $\quad$ **for** $j = 1$ to $sent\_size$ **do**
7: $\quad\quad LABEL \leftarrow w_j$
8: $\quad\quad PID \leftarrow j$
9: $\quad\quad SID \leftarrow i$
10: $\quad\quad$ **if** $ExistsNode(G, LABEL)$ **then**
11: $\quad\quad\quad v_j \leftarrow GetExistingNode(G, LABEL)$
12: $\quad\quad\quad PRI_{v_j} \leftarrow PRI_{v_j} \cup (SID, PID)$
13: $\quad\quad$ **else**
14: $\quad\quad\quad v_j \leftarrow CreateNewNode(G, LABEL)$
15: $\quad\quad\quad PRI_{v_j} \leftarrow (SID, PID)$
16: $\quad\quad$ **end if**
17: $\quad\quad$ **if not** $ExistsEdge(v_{j-1} \rightarrow v_j, G)$ **then**
18: $\quad\quad\quad AddEdge(v_{j-1} \rightarrow v_j, G)$
19: $\quad\quad$ **end if**
20: $\quad$ **end for**
21: **end for**

Our graph representation is closer to that used by Barzilay and Lee (Barzilay and Lee, 2003) for the task of paraphrasing, wherein each node in the graph represents a unique word. However, in their work, such a graph is used to identify regions of commonality and variability amongst similar sentences. Thus, the positional information is not required nor is it maintained. In contrast, we maintain positional information at each node as this is critical for the selection of candidate paths.

Algorithm **A1** outlines the steps involved in building an Opinosis-Graph. We start with a set of sentences relevant to a specific topic, which can be obtained in different ways depending on the application. For example, they may be all sentences related to the *battery life* of the *iPod Nano*. We denote these sentences as $Z = \{z_i\}_{i=1}^{n}$ where each z_i is a sentence containing part-of-speech (POS) annotations. (A1:4) Each $z_i \in Z$ is split into a set of *word units*, where each unit, w_j consists of a word and its corresponding POS annotation (e.g. "*service:nn*", "*good:adj*"). (A1:7-9) Each unique w_j will form a node, v_j, in the Opinosis-Graph, with w_j being the label. Also, since we only have one node per unique word unit, each node keeps track of all sentences that it is a part of using a *sentence identifier* (SID) along with its *position of occurrence* in that sentence (PID). (A1:10-16) Each node will thus carry a *Positional Reference Information* (PRI) which is a list of {SID:PID} pairs representing the node's membership in a sentence. (A1:17-19) The original structure of a sentence is recorded with the use of directed edges. Figure 1 shows a resulting Opinosis-Graph based on four sentences.

The Opinosis-Graph has some unique properties that are crucial in generating abstractive summaries. We highlight some of the core properties by drawing examples from Figure 1:

Property 1. *(**Redundancy Capture**). Highly redundant discussions are naturally captured by subgraphs.*

Figure 1 shows that although the phrase '*great device*' was mentioned in different parts of sentences (1) and (3), this phrase forms a relatively heavy sub-path in the resulting graph. This is a good indication of salience.

Property 2. *(**Gapped Subsequence Capture**). Existing sentence structures introduce **lexical links** that facilitate the discovery of new sentences or reinforce existing ones.*

The main point conveyed by sentences (2) and (3) in Figure 1 is that *calls drop frequently*. However, this is expressed in slightly different ways and is reflected in the resulting subgraph. Since sentence (2) introduces a *lexical link* between '*drop*' and '*frequently*', the word '*too*' can be ignored for sentence (3) as the same amount of information is retained. This is analogous to capturing a *repetitive gapped subsequence* where similar sequences with minor variations are captured. With this, the subgraph *calls drop frequently* can be considered redundant.

Property 3. *(**Collapsible Structures**). Nodes that resemble hubs are possibly collapsible.*

In Figure 1 we see that the subgraph '*the iPhone is*', is fairly heavy and the 'is' node acts like a

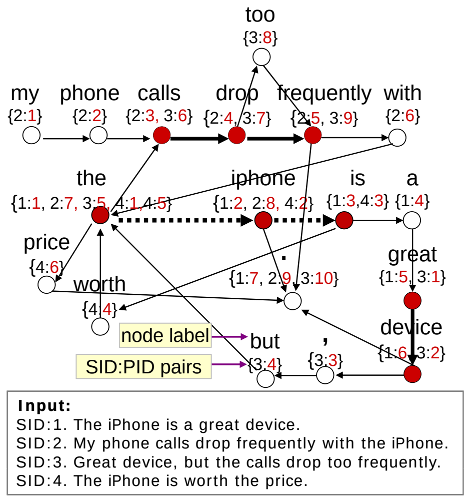


Figure 1: Sample *Opinosis-Graph*. Thick edges indicate salient paths.

'*hub*' where it connects to various other nodes. Such a structure is naturally captured by the Opinosis-Graph and is a good candidate for compression to generate a summary such as '*The iPhone is a great device and is worth the price*'. Also, certain word POS (e.g. linking verbs like 'is' and 'are') often carry hub-like properties that can be used in place of the outlink information.

3 Opinosis Summarization Framework

In this section, we describe a general framework for generating abstractive summaries using the Opinosis-Graph. We also describe our implementation of the components in this framework.

At a high level, we generate an abstractive summary by repeatedly searching the Opinosis graph for appropriate subgraphs that both encode a valid sentence (thus meaningful sentences) and have high redundancy scores (thus representative of the major opinions). The sentences encoded by these subgraphs would then form an abstractive summary.

Going strictly by the definition of true abstraction (Radev et al., 2002), our problem formulation is still more extractive than abstractive because the generated summary can only contain words that occur in the text to be summarized; our problem definition may be regarded as a word-level (finer granularity) extractive summarization. However, compared to the conventional sentence-level extractive summarization, our formulation has flavors of abstractive summarization wherein we have elements of *fusion* (combining extracted portions) and *compression* (squeezing out unimportant material from a sentence). Hence, the sentences in the generated summary are generally not the same as any original sentence. Such a "shallow" abstractive summarization problem is more tractable, enabling us to develop a general solution to the problem. We now describe each component in such a summarization framework.

3.1 Valid Path

A valid path intuitively refers to a path that corresponds to a meaningful sentence.

Definition 1. *(**Valid Start Node - VSN**). A node v_q is a valid start node if it is a natural starting point of a sentence.*

We use the positional information of a node to determine if it is a VSN. Specifically, we check if $Average(PID_{v_q}) \leq \sigma_{vsn}$, where σ_{vsn} is a parameter to be empirically set. With this, we only qualify nodes that tend to occur early on in a sentence.

Definition 2. *(**Valid End Node - VEN**). A node v_s is a valid end point if it completes a sentence.*

We use the natural ending points in the text to be summarized as hints to which node may be a valid end point of a path (i.e., a sentence). Specifically, a node is a *valid end node* if (1) the node is a punctuation such as *period* and *comma* or (2) the node is any coordinating conjunction (e.g., *'but'* and *'yet'*).

Definition 3. *(**Valid Path**). A path $W = \{v_q ... v_s\}$ is valid if it is connected by a set of directed edges such that (1) v_q is a **VSN**, (2) v_s is a **VEN**, and (3) W satisfies a set of well-formedness POS constraints.*

Since not every path starting with a VSN and ending at a VEN encodes a meaningful sentence, we further require a valid path to satisfy the following POS constraints (expressed in regular-expression) to ensure that a valid path encodes a well-formed sentence:

1. $.*(/nn)+.*(/vb)+.*(/jj)+.*$
2. $.*(/jj)+.*(/to)+.*(/vb).*$
3. $.*(/rb)*.*(/jj)+.*(/nn)+.*$
4. $.*(/rb)+.*(/in)+.*(/nn)+.*$

This also provides a way (if needed) for the application to generate only specific type of sentences like *comparative sentences* or *strictly opinionated sentences*. These rules are thus application specific.

3.2 Path Scoring

Intuitively, to generate an abstractive summary, we should select a valid path that can represent most of the redundant opinions well. We would thus favor a valid path with a high redundancy score.

Definition 4. *(**Path Redundancy**). Let $W = \{v_q ... v_s\}$ be a path from an Opinosis-Graph. The path redundancy of W, $r(q, s)$, is the number of overlapping sentences covered by this path, i.e.,*

$$r(q,s) = n_q \bar{\cap} n_{q+1} ... \bar{\cap} n_s,$$

where $n_i = PRI_{v_i}$ *and* $\bar{\cap}$ *is the intersection between two sets of SIDs such that the difference between the corresponding PIDs is no greater than* σ_{gap}*, and* $\sigma_{gap} > 0$ *is a parameter.*

Path redundancies provide good indication of how many sentences discuss something similar at each point in the path. The σ_{gap} parameter controls the maximum allowed gaps in discovering these redundancies. Thus, a common sentence X between nodes v_q and v_r, will be considered a valid intersect if $(PID_{v_{rx}} - PID_{v_{qx}}) \leq \sigma_{gap}$.

Based on path redundancy, we propose several ways to score a path for the purpose of selecting a good path to include in the summary:

1. $S_{basic}(W) = \frac{1}{|W|} \sum_{k=i+1,i}^{s} r(i,k)$
2. $S_{wt\_len}(W) = \frac{1}{|W|} \sum_{k=i+1,i}^{s} |v_i, v_k| * r(i,k)$
3. $S_{wt\_loglen}(W) = \frac{1}{|W|}(r(i,i+1) + \sum_{k=i+2,i+1}^{s} log_2|v_i, v_k| * r(i,k))$

v_i is the first node in the path being scored and v_s is the last node. $|v_i, v_k|$ is the length from node v_i to v_k. $|W|$ is the length of the entire path being scored. The S_{basic} scoring function scores a path purely based on the level of redundancy. One could also argue that high redundancy on a longer path is intuitively more valuable than high redundancy on a shorter path as the former would provide better coverage than the latter. This intuition is factored in by the S_{wt_len} and S_{wt_loglen} scoring functions where the level of *redundancy* is *weighted* by the *path length*. S_{wt_loglen} is similar to S_{wt_len} only that it scales down the path length so that it does not entirely dominate.

3.3 Collapsed paths

In some cases, paths in the Opinosis-Graph may be collapsible (as explained in Section 2). In such a case, the collapse operation is performed and then the path scores are computed. We will now explain a few concepts related to collapsible structures. Let $\widehat{W} = \{v_i ... v_k\}$ be a path from the Opinosis-Graph.

Definition 5. ***(Collapsible Node)****.* *Node* v_k *is a candidate for collapse if its POS is a* verb.

We only attempt to collapse nodes that are *verbs* due to the heavy usage of verbs in opinion text and the ease with which the structures can be combined to form a new sentence. However, as mentioned earlier other properties like the outlink information can be used to determine if a node is collapsible.

Definition 6. ***(Collapsed Candidates, Anchor)****.* *Let* v_k *be a collapsible node. The collapsed candidates of* v_k *(denoted by* $CC = \{cc_i\}_{i=1}^{m}$*) are the remaining paths after* v_k *in all the valid paths going through* $v_i ... v_k$*. The prefix* $v_i ... v_k$ *is called the anchor, denoted as* $C_{anchor} = \{v_i ... v_k\}$*. Each path* $\{v_i ... v_n\}$*, where* v_n *is the last node in each* $cc_i \in CC$*, is an individually valid path.*

C_{anchor}	CC	Connector
a. the sound quality is	cc_1 : really good cc_2 : clear	and
b. the iphone is	cc_1 : great cc_2 : expensive	but

Table 1: Example of anchors, collapsed candidates and suitable connectors

Table 1 shows a simplistic example of anchors and corresponding collapsed candidates. Once the anchor and collapsed candidates have been identified, the task is then to combine all of these to form a new sentence.

Definition 7. ***(Stitched Sentence)*** *A stitched sentence is one that combines* C_{anchor} *and* CC *to form a combined, logical sentence.*

We will now describe the stitching procedure that we use, by drawing examples from Table 1. Since we are dealing with verbs, C_{anchor} can be combined with the corresponding CC with commas to separate each $cc_i \in CC$ with one exception - the correct sentence connector has to be used for the last cc_i. For C_{anchor_a}, the phrases *really good* and *clear* can be connected by 'and' due to the same sentiment orientation. For C_{anchor_b}, the collapsed candidate phrases are well connected by the word 'but'. We use the existing Opinosis-Graph to determine the most appropriate connector. We do this by looking at all *coordinating conjunction* (e.g. 'but', 'yet') nodes (v_{cconj}) that are connected to the first node of the last collapsed candidate, cc_m. This would be the node labeled '*clear*' for C_{anchor_a} and '*expensive*' for C_{anchor_b}. We denote these nodes as v_{0,cc_m}. The v_{cconj}, with the highest *path redundancy* with v_{0,cc_m}, will be selected as the connector.

Definition 8. ***(Collapsed Path Score)*** *The final path score after the entire collapse operation is the average across path scores computed from* v_i *to the last node in each* $cc_i \in CC$*.*

The collapsed path score essentially involves computing the *path scores* of the individual sentences assuming that they are not collapsed and then averaging them.

3.4 Generation of summary

Once we can score all the valid paths as well as all the collapsed paths, the generation of an abstractive summary can be done in two steps: First, we rank all the paths (including the collapsed paths) in descending order of their scores. Second, we eliminate duplicated (or extremely similar) paths by using a similarity measure (in our experiments, we used Jaccard). We then take the top few remaining paths as the generated summary, with the number of paths to be chosen controlled by a parameter σ_{ss}, which represents summary size.

Although conceptually we enumerate all the valid paths, in reality we can use a redundancy score threshold, σ_r to prune many non-promising paths. This is reasonable because we are only interested in paths with high redundancy scores.

4 Summarization Algorithm

Algorithms **A2** and **A3** describe the steps involved in Opinosis Summarization. A2 is the starting point of the Opinosis Summarization and A3 is a subroutine where path finding takes place, invoked from within A2.

Algorithm 2 (A2): $OpinosisSummarization(Z)$

1: **Input:** Topic related sentences to be summarized: $Z = \{z_i\}_{i=1}^{n}$
2: **Output:** $\mathcal{O} = $ {Opinosis Summaries}
3: $g \leftarrow OpinosisGraph(Z)$
4: $node\_size \leftarrow SizeOf(g)$
5: **for** $j = 1$ to $node\_size$ **do**
6:   **if** $VSN(v_j)$ **then**
7:     $pathLen \leftarrow 1$
8:     $score \leftarrow 0$
9:     $cList \leftarrow CreateNewList()$
10:     **Traverse**$(cList, v_j, score, PRI_{v_j}, label_{v_j}, pathLen)$
11:     $candidates \leftarrow \{candidates \cup cList\}$
12:   **end if**
13: **end for**
14: $\mathcal{C} \leftarrow EliminateDuplicates(candidates)$
15: $\mathcal{C} \leftarrow SortByPathScore(\mathcal{C})$
16: **for** $i = 1$ to σ_{ss} **do**
17:   $\mathcal{O} = \{\mathcal{O} \cup PickNextBestCandidate(\mathcal{C})\}$
18: **end for**

(A2:3) Opinosis Summarization starts with the construction of the Opinosis-Graph, described in detail in Section 2. This is followed by the *depth first* traversal of this graph to locate *valid paths* that become *candidate summaries*. (A2:6-12) To achieve this, each node v_j in the Opinosis-Graph is examined to determine if it is a *VSN* and, if it is, path finding will start from this node by invoking subroutine A3. A3 takes the following as input: *list* - a list to hold candidate summaries; v_i - the node to continue traversal from; *score* - the accumulated path score; $PRI_{overlap}$ - the intersect between PRIs of all nodes visited so far (see Definition 4); *sentence* - the summary sentence formed so far; *len* - the current path length. (A2:7-10) Before invoking A3 from A2, the path length is set to '1', path score is set to '0' and a new list is created to store candidate summaries generated from node v_j. (A2:11) All candidate summaries generated from v_j will be stored in a common pool of candidate summaries.

Algorithm 3 (A3): $Traverse(...)$

1: **Input:** $list, v_k \subseteq V, score, PRI_{overlap}, sentence, len$
2: **Output:** A set of candidate summaries
3: $redundancy \leftarrow SizeOf(PRI_{overlap})$
4: **if** $redundancy \geq \sigma_r$ **then**
5:   **if** $VEN(v_k)$ **then**
6:     **if** $ValidSentence(sentence)$ **then**
7:       $finalScore \leftarrow \frac{score}{len}$
8:       $AddCandidate(list, sentence, finalScore)$
9:     **end if**
10:   **end if**
11:   **for** $v_n \in Neighbors_{v_k}$ **do**
12:     $PRI_{new} \leftarrow PRI_{overlap} \;\tilde{\cap}\; PRI_{v_n}$
13:     $redundancy \leftarrow SizeOf(PRI_{new})$
14:     $newSent \leftarrow Concat(sentence, label_{v_n})$
15:     $L \leftarrow len + 1$
16:     $newScore \leftarrow score + PathScore(redundancy, L)$
17:     **if** $Collapsible(v_n)$ **then**
18:       $C_{anchor} \leftarrow newSent$
19:       $tmp \leftarrow CreateNewList()$
20:       **for** $v_x \in Neighbors_{v_n}$ **do**
21:         **Traverse**$(tmp, v_x, 0, PRI_{new}, label_{v_x}, L)$
22:         $CC \leftarrow EliminateDuplicates(tmp)$
23:         $CCPathScore \leftarrow AveragePathScore(CC)$
24:         $finalScore \leftarrow newScore + CCPathScore$
25:         $stitchedSent \leftarrow Stitch(C_{anchor}, CC)$
26:         $AddCandidate(list, stitchedSent, finalScore)$
27:       **end for**
28:     **else**
29:       **Traverse**$(list, v_n, newScore, PRI_{new}, newSent, L)$
30:     **end if**
31:   **end for**
32: **end if**

(A3:3-4) Algorithm A3 starts with a check to ensure that the minimum path redundancy requirement is satisfied (see definition 4). For the very first node sent from A2, the path redundancy is the size of the raw PRI. (A3:5-10) If the redundancy requirement is satisfied, a few checks are done to determine if a *valid path* has been found. If it has, then the resulting sentence and its final score are added to the list of candidate summaries.

(A3:11-31) Traversal proceeds recursively through the exploration of all neighboring nodes of the current node, v_k. (A3:12-16) For every neighboring node, v_n the *PRI* overlap information, path length, summary sentence and path score are updated before the next recursion. (A3:29) If a v_n is not collapsible, then a regular traversal takes place. (A3:17-27) However, if v_n is collapsible, the updated sentence in A3:14, will now serve as an *anchor* in A3:18. (A3:21) A3 will then attempt to start a recursive traversal from all neighboring nodes of v_n in order to find corresponding collapsed candidates. (A3:22-26) After this, duplicates are eliminated from the *collapsed candidates* and the *collapsed path score* is computed. The resulting *stitched sentence* and its *final score* are then added to the original list of candidate summaries.

(A2:14-18) Once all paths have been explored for candidate generation, duplicate candidates are removed and the remaining are sorted in descending order of their path scores. The best σ_{ss} candidates are 'picked' as final Opinosis summaries.

5 Experimental Setup

We evaluate this abstractive summarization task using reviews of *hotels*, *cars* and various *products*[1]. Based on these reviews, 2 humans were asked to construct 'opinion seeking' queries which would consist of an *entity name* and a *topic of interest*. Example of such queries are: *Amazon Kindle:buttons*, *Holiday Inn, Chicago: staff*, and so on. We compiled a set of 51 such queries. We create one review document per query by collecting all review sentences that contain the query words for the given entity. Each review document thus consists of a set of unordered, redundant review sentences related to the query. There are approximately 100 sentences per review document.

We use ROUGE (Lin, 2004b) to quantitatively assess the agreement of Opinosis summaries with human composed summaries. ROUGE is based on an n-gram co-occurrence between machine summaries and human summaries and is a widely accepted standard for evaluation of summarization tasks. In our experiments, we use ROUGE-1, ROUGE-2 and ROUGE-SU4 measures. ROUGE-1 and ROUGE-2 have been shown to have most correlation with human summaries (Lin and Hovy, 2003) and higher order ROUGE-N scores ($N > 1$) estimate the fluency of summaries.

We use multiple reference (human) summaries in our evaluation since it can achieve better correlation with human judgment (LIN, 2004a). We leverage Amazon's Online Workforce[2] to get 5 different human workers to summarize each review document. The workers were asked to be concise and were asked to summarize the major opinions in the review document presented to them. We manually reviewed each set of reference summaries and dropped summaries that had little or no correlation with the majority. This left us with around 4 reference summaries for each review document.

To allow performance comparison between humans, Opinosis and the baseline method, we implemented a Jackknifing procedure where, given K references, the ROUGE score is computed over K sets of K-1 references. With this, average human performance is computed by treating each reference summary as a 'system' summary, computing ROUGE scores over the remaining K-1 reference summaries.

Due to the limited work in abstractive summarization, no natural baseline could be used for comparison. The existing work in this area is mostly domain dependent and requires too much manual effort (explained in Section 1). The next best baseline is to use a state of the art extractive method. Thus, we use MEAD (Radev et al., 2000) as our baseline. MEAD is an extractive summarizer based on cluster centroids. It uses a collection of the most important words from the whole cluster to select the best sentences for summarization. By default, the scoring of sentences in MEAD is based on 3 parameters - *minimum sentence length*, *centroid*, and *position in text*. MEAD was ideal for our task because a good summary in our case would be one that could capture the most essential information. This is exactly what centroid-based summarization aims to achieve. Also, since the *position in text* parameter is irrelevant in our case, we could easily turn this off with MEAD.

We introduce a **readability test** to understand if Opinosis summaries are in fact readable. Suppose we have N sentences from a system-generated summary and M sentences from corresponding human summaries. We mix all these sentences and then ask a human assessor to pick at most N sentences that are *least readable* as the prediction of system summary.

$$readability(O) = 1 - \frac{\#CorrectPick}{N}$$

If the human assessor often picks out system generated summaries as being least readable, then the readability of system summaries is poor. If not, then the system generated summaries are no different from human summaries.

6 Results

The baseline method (MEAD) selects 2 most representative sentences as summaries. To give a fair comparison, we fix the Opinosis summary size, $\sigma_{ss} = 2$. We also fix $\sigma_{vsn} = 15$. The best Opinosis configuration with $\sigma_{ss} = 2$ and $\sigma_{vsn} = 15$ is called Opinosis$_{best}$ ($\sigma_{gap} = 4, \sigma_r = 2$, S_{wt_loglen}). ROUGE scores reported are with the use of *stemming* and *stopword removal*.

Performance comparison between humans, Opinosis and baseline. Table 2 shows the performance comparison between humans, Opinosis$_{best}$ and the baseline method. First, we see that the baseline method has very high recall scores compared to Opinosis. This is because extractive methods that just 'select' sentences tend to be much longer resulting in higher recall. However, these summaries tend to carry information that may not be significant and is clearly reflected by the poor

[1] Reviews collected from Tripadvisor, Amazon, Edmunds
[2] https://www.mturk.com

	ROUGE-1	ROUGE-2	ROUGE-SU4	Avg # Words
Recall				
Human	0.3184	**0.1106**	0.1293	17
Opinosis	0.2831	0.0853	0.0851	15
Baseline	**0.4932**	0.1058	**0.2316**	75
Precision				
Human	0.3434	0.1210	0.1596	17
Opinosis	**0.4482**	**0.1416**	**0.2261**	15
Baseline	0.0916	0.0184	0.0102	75
F-score				
Human	0.3088	**0.1069**	**0.1142**	17
Opinosis	**0.3271**	0.0998	0.1027	15
Baseline	0.1515	0.0308	0.0189	75

Table 2: Performance comparison between Humans, Opinosis$_{best}$ and Baseline.

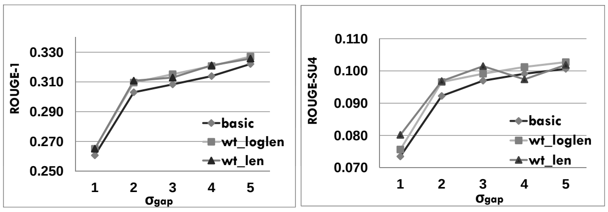


Figure 2: ROUGE scores (f-measure) at different levels of σ_{gap}, $\sigma_r = 2$.

precision scores.

Next, we see that humans have reasonable agreement amongst themselves given that these are independently composed summaries. This agreement is especially clear with the ROUGE-2 recall score where the recall is better than Opinosis but comparable to the baseline even though the summaries are much shorter. It is also clear that Opinosis is closer in performance to humans than to the baseline method. The recall scores of Opinosis summaries are slightly lower than that achieved by humans, while the precision scores are higher (Wilcoxon test shows that the increase in precision is statistically more significant than the decrease in recall). In terms of f-scores, Opinosis has the best ROUGE-1 score and its ROUGE-2 and ROUGE-SU4 scores are comparable with human performance. The baseline method has the lowest f-scores. The difference between the f-scores of Opinosis and that of humans is statistically insignificant.

Comparison of scoring functions. Next, we look into the performance of the three scoring functions, S_{basic}, S_{wt_len} and S_{wt_loglen} described in Section 3. Figure 2 shows ROUGE scores of these scoring methods at varying levels of σ_{gap}. First,

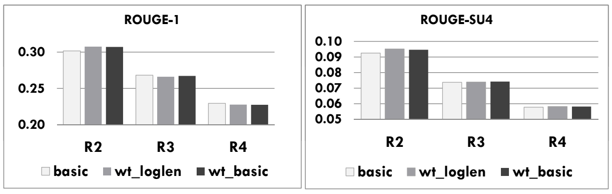


Figure 3: ROUGE scores (f-measure) at different levels of σ_r averaged across $\sigma_{gap} \in [1, 5]$

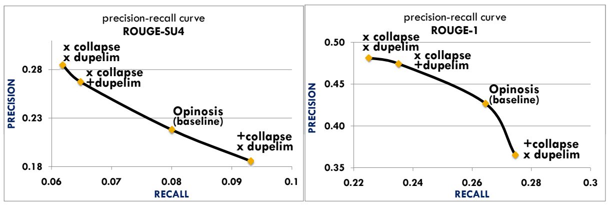


Figure 4: Precision-Recall comparison with different Opinosis features turned off.

it can be observed that S_{wt_basic} which does not use *path length* information, performs the worst. This is due to the effect of heavily favoring redundant paths over longer but reasonably redundant ones that can provide more coverage. We also see that S_{wt_len} and S_{wt_loglen} are similar in performance with S_{wt_loglen} marginally outperforming S_{wt_len} when $\sigma_{gap} > 2$. Since S_{wt_len} uses the raw *path length* in its scoring function, it may be inflating the path scores of long but insignificant paths. S_{wt_loglen} scales down the path length, thus providing a reasonable tradeoff between redundancy and the length of the selected path. The three scoring functions are not influenced by different levels of σ_r as shown in Figure 3.

Effect of gap setting (σ_{gap}). Now, we will examine the effect of σ_{gap} on the generated summaries. Based on Figure 2, we see that setting σ_{gap}=1 yields in relatively low performance. This is because σ_{gap}=1 implies immediate adjacency between the PIDs of two nodes and such strict adjacency enforcements prevent redundancies from being discovered. When σ_{gap} is increased to 2, there is a big jump in performance, after which improvements are observed in smaller amounts. A very large gap setting could increase the possibility of generating ill-formed sentences, thus we recommend that σ_{gap} is set between 2-5.

Effect of redundancy requirement (σ_r) . Figure 3 shows the ROUGE scores at different levels of σ_r. It is clear that when $\sigma_r > 2$, the quality of summaries is negatively impacted. Since we only have about 100 sentences per review document, $\sigma_r > 2$ severely restricts the number of paths that can be explored, yielding in lower ROUGE scores. Since the scoring function can account for the level of redundancy, σ_r should be set according to the size of the input data. For our dataset, $\sigma_r = 2$ was ideal.

"About food at Holiday Inn, London"	"What is free at Bestwestern Inn, San Francisco"
Human summaries: **[1]** Food was excellent with a wide range of choices and good services. **[2]** The food is good, the service great. Very good selection of food for breakfast buffet.	**Human summaries:** **[1]** There is free WiFi internet access available in all the rooms.. From 5-6 p.m. there is free wine tasting and appetizers available to all the guests. **[2]** Evening wine reception and free coffee in the morning. Free internet, free parking and free massage.
Opinosis abstractive summary: The food was excellent, good and delicious. Very good selection of food.	**Opinosis abstractive summary:** Free wine reception in evening. Free coffee and biscotti and wine.
Baseline extractive summary: Within 200 yards of leaving the hotel and heading to the Tube Station you have a number of fast food outlets, highstreet Restautants, Pastry shops and supermarkets, so if you did wish to live in your hotel room for the duration of your stay, you could do.......	**Baseline extractive summary:** The free wine and nibbles served between 5pm and 6pm were a lovely touch. There's free coffee, teas at breakfast time with little biscotti and, best of all, from 5 till 6pm you get a free wine 'tasting' reception which, as long as you don't take……

Figure 5: Sample results comparing Opinosis summaries with human and baseline summaries.

Effect of collapsed structures and duplicate elimination. So far, it has been assumed that all features used in Opinosis are required to generate reasonable summaries. To test this hypothesis, we use Opinosis$_{best}$ as a baseline and then we turn off different features of Opinosis. We turn off the *duplicate elimination feature*, then the *collapsible structure feature*, and finally *both*. Figure 4 shows the resulting precision-recall curve. From this graph, we see that without duplicate elimination and when collapsing is turned off, the precision is highest but recall is lowest. No collapsing implies shorter sentences and thus lower recall, which is clearly reflected in Figure 4. On top of this, if duplicates are allowed, the overall information coverage is low, further affecting the recall. Notice that the presence of duplicates with the collapse feature turned on results in very high recall (even higher than the baseline). This is caused by the presence of similar phrases that were not eliminated from the collapsed candidates, resulting in long sentences that artificially boost recall. The Opinosis baseline which uses duplicate elimination and the collapsible structure feature, offers a reasonable tradeoff between precision and recall.

Readability of Summaries. To test the readability of Opinosis summaries, we conducted a *readability test* (described in Section 5) using summaries generated from Opinosis$_{best}$. A human assessor picked the 2 least readable sentences from each of the 51 test sets (based on 51 summaries). Collectively, there were 565 sentences out of which 102 were Opinosis generated. Out of these, the human assessor picked only 34 of the sentences as being least readable, resulting in an average readability score of 0.67. This shows that more than 60% of the generated sentences are indistinguishable from human composed sentences. Of the 34 sentences with problems, 11 contained *no information* or were *incomprehensible*, 12 were *incomplete* possibly due to false positives when the sentence validity check was done, and 8 had *conflicting information* such as '*the hotel room is clean and dirty*'. This happens due to mixed feelings about the same topic and can be resolved using sentiment analysis. The remaining 3 sentences were found to contain *poor grammar*, possibly caused by the gaps allowed in finding redundant paths.

Sample Summaries. Finally, in Figure 5 we show two sample summaries on two different topics. Notice that the Opinosis summaries are concise, fairly well-formed and have closer resemblance to human summaries than to the baseline summaries.

7 Conclusion

In this paper, we described a novel summarization framework (Opinosis) that uses textual graphs to generate abstractive summaries of highly redundant opinions. Evaluation results on a set of review documents show that Opinosis summaries have better agreement with human summaries compared to the baseline extractive method. The Opinosis summaries are concise, reasonably well-formed and communicate essential information. Our readability test shows that more than 60% of the generated sentences are no different from human composed sentences.

Opinosis is a flexible framework in that many of its modules can be easily improved or replaced with other suitable implementation. Also, since Opinosis is domain independent and relies on minimal external resources, it can be used with any corpus containing high amounts of redundancies.

Our graph representation naturally ensures the coherence of a summary, but such a graph emphasizes too much on the surface order of words. As a result, it cannot group sentences at a deep semantic level. To address this limitation, we can use a similar idea to overlay parse trees and this would be a very interesting future research.

8 Acknowledgments

We thank the anonymous reviewers for their useful comments. This paper is based upon work supported in part by an IBM Faculty Award, an Alfred P. Sloan Research Fellowship, an AFOSR MURI Grant FA9550-08-1-0265, and by the National Science Foundation under grants IIS-0347933, IIS-0713581, IIS-0713571, and CNS-0834709.

References

[Barzilay and Lee2003] Barzilay, Regina and Lillian Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 16–23, Morristown, NJ, USA. Association for Computational Linguistics.

[DeJong1982] DeJong, Gerald F. 1982. An overview of the FRUMP system. In Lehnert, Wendy G. and Martin H. Ringle, editors, *Strategies for Natural Language Processing*, pages 149–176. Lawrence Erlbaum, Hillsdale, NJ.

[Erkan and Radev2004] Erkan, Günes and Dragomir R. Radev. 2004. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479.

[Finley and Harabagiu2002] Finley, Sanda Harabagiu and Sanda M. Harabagiu. 2002. Generating single and multi-document summaries with gistexter. In *Proceedings of the workshop on automatic summarization*, pages 30–38.

[Jing and McKeown2000] Jing, Hongyan and Kathleen R. McKeown. 2000. Cut and paste based text summarization. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 178–185, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Lerman et al.2009] Lerman, Kevin, Sasha Blair-Goldensohn, and Ryan Mcdonald. 2009. Sentiment summarization: Evaluating and learning user preferences. In *12th Conference of the European Chapter of the Association for Computational Linguistics (EACL-09).*

[Lin and Hovy2003] Lin, Chin-Yew and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proc. HLT-NAACL*, page 8 pages.

[LIN2004a] LIN, Chin-Yew. 2004a. Looking for a few good metrics : Rouge and its evaluation. *proc. of the 4th NTCIR Workshops, 2004*.

[Lin2004b] Lin, Chin-Yew. 2004b. Rouge: a package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out (WAS 2004), Barcelona, Spain*.

[Lu et al.2009] Lu, Yue, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *18th International World Wide Web Conference (WWW2009)*, April.

[Mihalcea and Tarau2004] Mihalcea, R. and P. Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing*, July.

[Pang and Lee2004] Pang, Bo and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, pages 271–278.

[Pang et al.2002] Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86.

[Radev and McKeown1998] Radev, DR and K. McKeown. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469–500.

[Radev et al.2000] Radev, Dragomir, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *In ANLP/NAACL Workshop on Summarization*, pages 21–29.

[Radev et al.2002] Radev, Dragomir R., Eduard Hovy, and Kathleen McKeown. 2002. Introduction to the special issue on summarization.

[Saggion and Lapalme2002] Saggion, Horacio and Guy Lapalme. 2002. Generating indicative-informative summaries with sumum. *Computational Linguistics*, 28(4):497–526.

[Snyder and Barzilay2007] Snyder, Benjamin and Regina Barzilay. 2007. Multiple aspect ranking using the good grief algorithm. In *In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL*, pages 300–307.

[Titov and Mcdonald2008] Titov, Ivan and Ryan Mcdonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of ACL-08: HLT*, pages 308–316, Columbus, Ohio, June. Association for Computational Linguistics.

EMDC: A Semi-supervised Approach for Word Alignment

Qin Gao
Language Technologies Institute
Carnegie Mellon University
qing@cs.cmu.edu

Francisco Guzman
Centro de Sistemas Inteligentes
Tecnológico de Monterrey
guzmanhe@gmail.com

Stephan Vogel
Language Technologies Institute
Carnegie Mellon University
stephan.vogel@cs.cmu.edu

Abstract

This paper proposes a novel semi-supervised word alignment technique called EMDC that integrates discriminative and generative methods. A discriminative aligner is used to find high precision partial alignments that serve as constraints for a generative aligner which implements a constrained version of the EM algorithm. Experiments on small-size Chinese and Arabic tasks show consistent improvements on AER. We also experimented with moderate-size Chinese machine translation tasks and got an average of 0.5 point improvement on BLEU scores across five standard NIST test sets and four other test sets.

1 Introduction

Word alignment is a crucial component in statistical machine translation (SMT). From a Machine Learning perspective, the models for word alignment can be roughly categorized as generative models and discriminative models. The widely used word alignment tool, i.e. GIZA++ (Och and Ney, 2003), implements the well-known IBM models (Brown et al., 1993) and the HMM model (Vogel et al., 1996), which are generative models. For language pairs such as Chinese-English, the word alignment quality is often unsatisfactory. There has been increasing interest on using manual alignments in word alignment tasks, which has resulted in several discriminative models. Ittycheriah and Roukos (2005) proposed to use only manual alignment links in a maximum entropy model, which is considered supervised. Also, a number of semi-supervised word aligners have been proposed (Taskar et al., 2005; Liu et al., 2005; Moore, 2005; Blunsom and Cohn, 2006; Niehues and Vogel, 2008). These methods use held-out manual alignments to tune weights for discriminative models, while using the model parameters, model scores or alignment links from unsupervised word aligners as features. Callison-Burch et. al. (2004) proposed a method to interpolate the parameters estimated by sentence-aligned and word-aligned corpus. Also, there are recent attempts to combine multiple alignment sources using alignment confidence measures so as to improve the alignment quality (Huang, 2009).

In this paper, the question we address is whether we can jointly improve discriminative models and generative models by feeding the information we get from the discriminative aligner back into the generative aligner. Examples of this line of research include Model 6 (Och and Ney, 2003) and the EMD training approach proposed by Fraser and Marcu (2006) and its extension called LEAF aligner (Fraser and Marcu, 2007). These approaches use labeled data to tune additional parameters to weight different components of the IBM models such as the lexical translation model, the distortion model and the fertility model. These methods are proven to be effective in improving the quality of alignments. However, the discriminative training in these methods is restricted in using the model components of generative models, in other words, incorporating new features is difficult.

Instead of using discriminative training methods to tune the weights of generative models, in this paper we propose to use a discriminative word aligner to produce reliable constraints for the EM algorithm. We call this new training scheme EMDC (**E**xpectation-**M**aximization-**D**iscrimination-**C**onstraint). The methodology can be viewed as a variation of bootstrapping. It enables the generative models to interact with discriminative models at the data level instead of the model level. Furthermore, with a discriminative

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 349–357,
Beijing, August 2010

word aligner that uses generative word aligner's output as features, we create a feedback loop that can iteratively improve the quality of both aligners. The major contributions of this paper are: 1) The EMDC training scheme, which ties the generative and discriminative aligners together and enables future research on integrating other discriminative aligners. 2) An extended generative aligner based on GIZA++ that allows to perform constrained EM training.

In Section 2, we present the EMDC training scheme. Section 3 provides details of the constrained EM algorithm. In Section 4, we introduce the discriminative aligner and link filtering. Section 5 provides the experiment set-up and the results. Section 6 concludes the paper.

2 EMDC Training Scheme

The EMDC training scheme consists of three parts, namely **EM**, **D**iscrimination, and **C**onstraints. As illustrated in Figure 1, a large unlabeled training set is first aligned with a generative aligner (GIZA++ for the purpose of this paper). The generative aligner outputs the model parameters and the Viterbi alignments for both source-to-target and target-to-source directions. Afterwards, a discriminative aligner (we use the one described in (Niehues and Vogel, 2008)), takes the lexical translation model, fertility model and Viterbi alignments from both directions as features, and is tuned to optimize the AER on a small manually aligned tuning set. Afterwards, the alignment links generated by the discriminative aligner are filtered according to their likelihood, resulting in a subset of links that has high precision and low recall. The next step is to put these high precision alignment links back into the generative aligner as constraints. A conventional generative word aligner does not support this type of constraints. Thus we developed a constrained EM algorithm that can use the links from a partial alignment as constraints and estimate the model parameters by marginalizing likelihoods.

After the constrained EM training is performed, we repeat the procedure and put the *updated* generative models and Viterbi alignment back into the discriminative aligner. We can either fix the number of iterations, or stop the procedure when the gain on AER of a small held-out test set drops below a threshold.

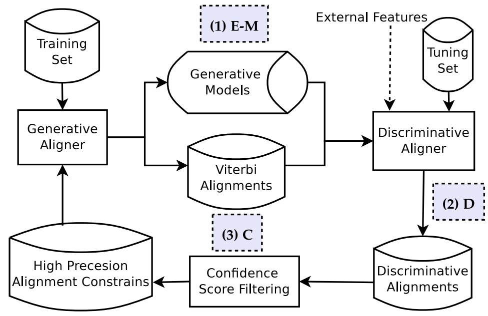


Figure 1: Illustration of EMDC training scheme

The key components for the system are:

1. A generative aligner that can make use of reliable alignment links as constraints and improve the models/alignments.
2. A discriminative aligner that outputs confidence scores for alignment links, which allows to obtain high-precision-low-recall alignments.

While in this paper we derive the reliable links by filtering the alignment generated by a discriminative aligner, such partial alignments may be obtained from other sources as well: manual alignments, specific named entity aligner, noun-phrase aligner, etc.

As we mentioned in Section 1, the discriminative aligner is not restricted to use features parameters of generative models and Viterbi alignments. However, including the features from generative models is required for iterative training, because the improvement on the quality of these features can in turn improve the discriminative aligner. In our experiments, the discriminative aligner makes heavy use of the Viterbi alignment and the model parameters from the generative aligner. Nonetheless, one can easily replace the discriminative aligner or add new features to it without modifying the training scheme. The open-ended property of the training scheme makes it a promising method to integrate different aligners.

In the next two sections, we will describe the key components of this framework in detail.

3 Constrained EM algorithm

In this section we will briefly introduce the constrained EM algorithm we used in the experiment, further details of the algorithm can be found in (Gao et al., 2010).

The IBM Models (Brown et al., 1993) are a series of generative models for word alignment. GIZA++ (Och and Ney, 2003), the most widely used implementation of IBM models and HMM (Vogel et al., 1996), employs EM algorithm to estimate the model parameters. For simpler models such as Model 1 and Model 2, it is possible to obtain sufficient statistics from all possible alignments in the E-step. However, for fertility-based models such as Models 3, 4, and 5, enumerating all possible alignments is NP-complete. To overcome this limitation, GIZA++ adopts a greedy hill-climbing algorithm, which uses simpler models such as HMM or Model 2 to generate a "center alignment" and then tries to find better alignments among its neighbors. The neighbors of an alignment $a_1^J = [a_1, a_2, \cdots, a_J]$ with $a_j \in [0, I]$ are defined as alignments that can be generated from a_1^J by one of the following two operators:

1. The move operator $m_{[i,j]}$, that changes $a_j := i$, i.e. arbitrarily sets word f_j in the target sentence to align to the word e_i in source sentence;
2. The swap operator $s_{[j_1,j_2]}$ that exchanges a_{j_1} and a_{j_2}.

The algorithm will update the center alignment as long as a better alignment can be found, and finally outputs a local optimal alignment. The neighbor alignments of the final center alignment are then used in collecting the counts for the M-Step. Och and Ney (2003) proposed a fast implementation of the hill-climbing algorithm that employs two matrices, i.e. Moving Matrix $M_{I\times J}$ and Swapping Matrix $S_{J\times J}$. Each cell of the matrices stores the value of likelihood difference after applying the corresponding operator.

We define a partial alignment constraint of a sentence pair (f_1^J, e_1^I) as a set of links: $\alpha_I^J = \{(i,j)|0 \leq i < I, 0 \leq j < J\}$. Given a set of constraints, an alignment $a_1^J = [a_1, a_2, \cdots, a_j]$ on the sentence pair f_1^J, e_1^I, the translation probability of $Pr(f_1^J|e_1^I)$ will be zero if the alignment is inconsistent with the constraints. Constraints $(0,j)$ or $(i,0)$ are used to explicitly represent that word f_j or e_i is aligned to the empty word.

Under the assumptions of the IBM models, there are two situations that a_1^J is inconsistent with α_I^J:

1. Target word misalignment: The IBM models assume that one target word can only be aligned to one source word. Therefore, if the target word f_j aligns to a source word e_i, while the constraint α_I^J suggests f_j should be aligned to $e_{i'}$, the alignment violates the constraint and thus is considered inconsistent.
2. Source word to empty word misalignment: if a source word is aligned to the empty word, it cannot be aligned to any concrete target word.

However, the partial alignments, which allow n-to-n alignments, may already violate the 1-to-n alignment restriction of the IBM models. In these cases, we relax the condition in situation 1 that if the alignment link a_{j^*} is consistent with any one of the conflicting target-to-source constraints, it will be considered consistent. Also, we arbitrarily assign the source word to empty word constraints higher priorities than other constraints, because unlike situation 1, it does not have the problem of conflicting with other constraints.

3.1 Constrained hill-climbing algorithm

To ensure that resulting center alignment be consistent with the constraints, we need to split the hill-climbing algorithm into two stages: 1) optimize towards the constraints and 2) optimize towards the optimal alignment under the constraints.

From a seed alignment, we first move the alignment towards the constraints by choosing a move or swap operator that:

1. produces the alignment that has the highest likelihood among alignments generated by other operators,
2. eliminates at least one inconsistent link.

We iteratively update the alignment until no other inconsistent link can be removed. The algorithm implies that we force the seed alignment to be closer to the constraints while trying to find the best consistent alignment. Figure 2 demonstrates the idea, given the constraints shown in (a), and the seed alignment shown as solid links in (b), we

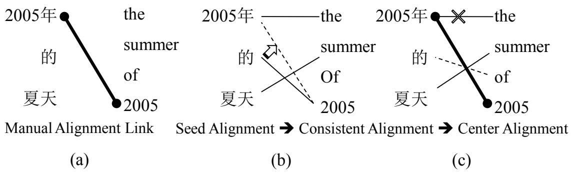


Figure 2: Illustration of Algorithm 1

move the inconsistent link to the dashed link by a move operation.

After we find the consistent alignment, we proceed to optimize towards the optimal alignment under the constraints. The algorithm sets the value of the cells in moving/swapping matrices to negative if the corresponding operators will lead to an inconsistent alignment. The moving matrix needs to be processed only once, whereas the swapping matrix needs to be updated every iteration, since once the alignment is updated, the possible violations will also change.

If a source word e_i is aligned to the empty word, we set $M_{i,j} = -1, \forall j$. The swapping matrix does not need to be modified in this case because the swapping operator will not introduce new links.

Because the cells that can lead to violations are set to negative, the operators will never be picked when updating the center alignments. This ensures the consistency of the final center alignment.

3.2 Count Collection

After finding the center alignment, we need to collect counts from neighbor alignments so that the M-step can normalize the counts to produce the model parameters for the next step. In this stage, we want to make sure all the inconsistent alignments in the neighbor set of the center alignment be ruled out from the sufficient statistics, i.e. have zero probability. Similar to the constrained hill climbing algorithm, we can manipulate the moving/swapping matrices to effectively exclude inconsistent alignments. Since the original count collection algorithm depends only on moving and swapping matrices, we just need to bypass all the cells which hold negative values, i.e. represent inconsistent alignments.

We can also view the algorithm as forcing the posteriors of inconsistent alignments to zero, and therefore increase the posteriors of consistent alignments. When no constraint is given, the algorithm falls back to conventional EM, and when all the alignments are known, the algorithm becomes fully supervised. And if the alignment quality can be improved if high-precision partial alignment links is given as constraints. In (Gao et al., 2010) we experimented with using a dictionary to generate such constraints, and in (Gao and Vogel, 2010) we experimented with manual word alignments from Mechanical Turk. And in this paper we try to use an alternative method that uses a discriminative aligner and link filtering to generate such constraints.

4 Discriminative Aligner and Link Filtering

We employ the CRF-based discriminative word aligner described in (Niehues and Vogel, 2008). The aligner can use a variety of knowledge sources as features, such as: the fertility and lexical translation model parameters from GIZA++, the Viterbi alignment from both source-to-target and target-to-source directions. It can also make use of first-order features which model the dependency between different links, the Parts-of-Speech tagging features, the word form similarity feature and the phrase features. In this paper we use all the features mentioned above except the POS and phrase features.

The aligner is trained using a belief-propagation (BP) algorithm, and can be optimized to maximize likelihood or directly optimize towards AER on a tuning set. The aligner outputs confidence scores for alignment links, which allows us to control the precision and recall rate of the resulting alignment. Guzman et al. (2009) experimented with different alignments produced by adjusting the filtering threshold for the alignment links and showed that they could get high-precision-low-recall alignments by having a higher threshold. Therefore, we replicated the confidence filtering procedures to produce the partial alignment constraints. Afterwards we iterate by putting the partial alignments back to the constrained word alignment algorithm described in section 3.

Although the discriminative aligner performs well in supplying high precision constraints, it does not model the null alignment explicitly.

	Num. of Sentences	Num. of Words *Source*	*Target*	Num. of Links
Ch-En	21,863	424,683	524,882	687,247
Ar-En	29,876	630,101	821,938	830,349

Table 1: Corpus statistics of the manual aligned corpora

	Threshold	P	R	AER
Ch-En	0.6	71.30	58.12	35.96
	0.7	75.24	54.03	37.11
	0.8	85.66	44.19	41.70
	0.9	93.70	37.95	45.98
Ar-En	0.6	72.35	59.87	34.48
	0.7	77.55	55.58	35.25
	0.8	80.07	50.89	37.77
	0.9	83.74	44.16	42.17

Table 2: The qualities of the constraints

Hence we are currently not able to provide source word to empty word alignment constraints which have been proven to be effective in improving the alignment quality in (Gao et al., 2010). Due to space limitation, please refer to: (Niehues and Vogel, 2008; Guzman et al., 2009) for further details of the aligner and link filtering, respectively.

5 Experiments

To validate the proposed training scheme, we performed two sets of experiments. First of all, we experimented with a small manually aligned corpus to evaluate the ability of the algorithm to improve the AER. The experiment was performed on Chinese to English and Arabic to English tasks. Secondly, we experimented with a moderate size corpus and performed translation tasks to observe the effects in translation quality.

5.1 Effects on AER

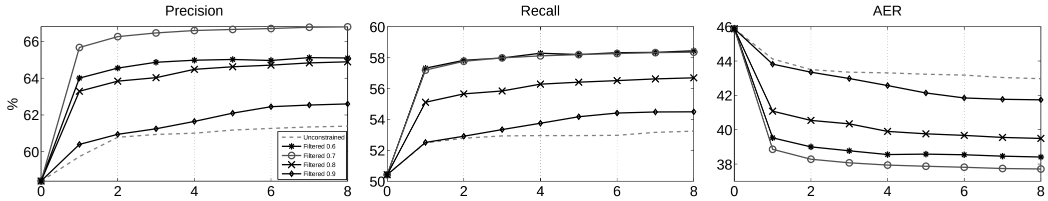
In order to measure the effects of EMDC in alignment quality, we experimented with Chinese-English and Arabic-English manually aligned corpora. The statistics of these sets are shown in Table 1. We split the data into two fragments, the first 100 sentences (Set A) and the remaining (Set B). We trained generative IBM models using the Set B, and tuned the discriminative aligner using the Set A. We evaluated the AER on Set B, but in any of the training steps the manual alignments of Set B were not used.

In each iteration of EDMC, we load the model parameters from the previous step and continue training using the new constraints. Therefore, it is important to compare the performance of continuous training against an unconstrained baseline, because variation in alignment quality could be attributed to either the effect of more training iterations or to the effect of semi-supervised training scheme. In Figures 3 and 4 we show the alignment quality for each iteration. Iteration 0 is the baseline, which comes from standard GIZA++ training[1]. The grey dash curves represent unconstrained Model 4 training, and the curves with start, circle, cross and diamond markers are constrained EM alignments with 0.6, 0.7, 0.8 and 0.9 filtering thresholds respectively. As we can see from the results, when comparing only the mono-directional trainings, the alignment qualities improve over the unconstrained training in all the metrics (precision, recall and AER). From Table 2, we observe that the quality of discriminative aligner also improved. Nonetheless, when we consider the heuristically symmetrized alignment[2], we observe mixed results. For instance, for the Chinese-English case we observe that AER improves over iterations, but this is the result of a increasingly higher recall rate in detriment of precision. Ayan and Dorr (2006) pointed out that *grow-diag-final* symmetrization tends to output alignments with high recall and low precision. However this does not fully explain the tendency we observed between iterations. The characteristics of the alignment modified by EDMC that lead to larger improvements in mono-directional trainings but a precision drop with symmetrization heuristics needs to be addressed in future work.

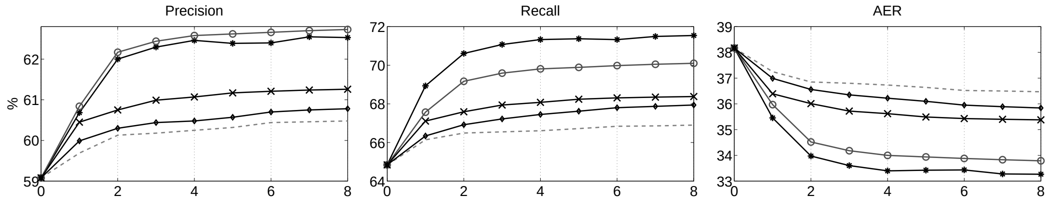
Another observation is how the filtering thresholds affect the results. As we can see in Table 3, for Chinese to English word alignment, the largest gain on the alignment quality is observed when the threshold was set to 0.8, while for Arabic to English, the threshold of 0.7 or 0.6 works better. Table 2 shows the precision, recall, and AER of the constraint links used in the constrained EM al-

[1] We run 5, 5, 3, 3 iterations of Model 1, HMM, Model 3 and Model 4 respectively.

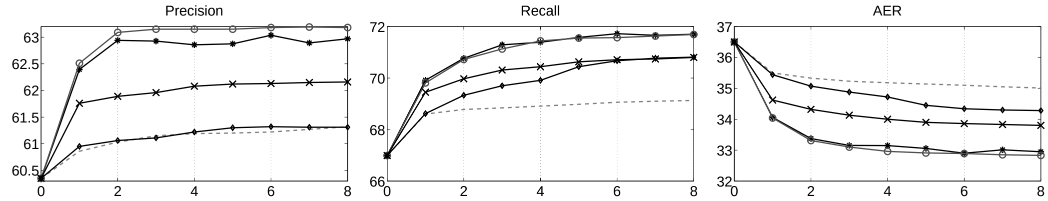
[2] We used grow-diag-final-and



(a) Arabic-English



(b) English-Arabic



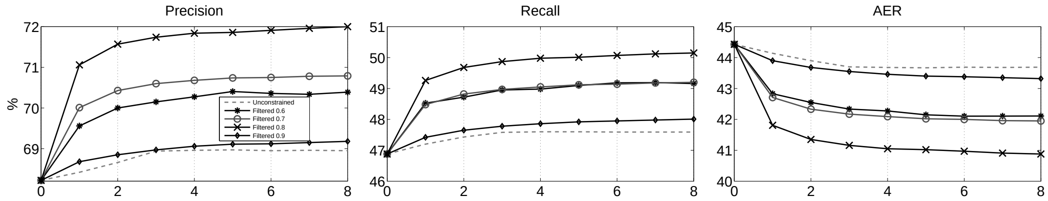
(c) Heuristically-symmetrized

Figure 3: Alignment qualities of each iteration for Arabic-English word alignment task. The grey dash curves represent unconstrained Model 4 training, and the curves with star, circle, cross and diamond markers are constrained EM alignments with 0.6, 0.7, 0.8 and 0.9 filtering thresholds respectively.

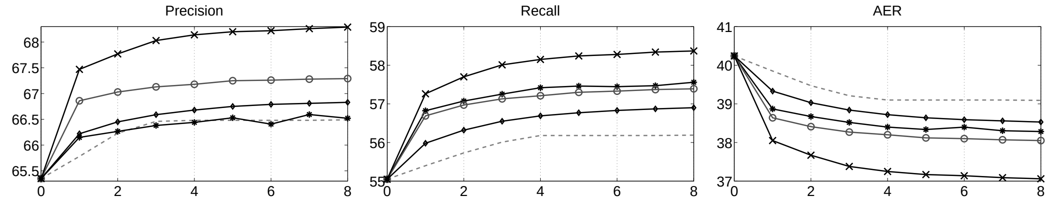
		Source-Target			**Target-Source**			**Heuristic**			**Discriminative**		
		P	R	AER	P	R	AER	P	R	AER	P	R	AER
Ch	BL	68.22	46.88	44.43	65.35	55.05	40.25	69.15	57.47	37.23	67.45	59.77	36.62
	NC	+0.73	+0.71	-0.74	+1.14	+1.14	-1.15	**+0.06**	+1.07	-0.66	+0.15	+0.64	-0.42
	0.6	+2.17	+2.28	-2.32	+1.17	+2.51	-1.97	-0.64	+2.65	-1.27	-0.39	+1.89	-0.87
	0.7	+2.57	+2.32	-2.48	+1.94	+2.34	-2.19	*-0.34*	+2.30	-1.20	*-0.28*	+1.60	-0.76
	0.8	**+3.78**	**+3.27**	**-3.55**	**+2.94**	**+3.32**	**-3.18**	*-0.52*	**+3.32**	**-1.70**	**+0.69**	**+0.14**	**-0.89**
	0.9	+0.98	+1.13	-1.11	+1.48	+1.85	-1.71	*-0.55*	+1.94	-0.90	*-0.58*	+1.45	-0.54
Ar	BL	58.41	50.42	45.88	59.08	64.84	38.17	60.35	66.99	36.50	68.93	63.94	33.66
	NC	+2.98	+2.92	-2.96	+1.40	+2.06	-1.70	+0.97	+2.14	-1.49	*-0.87*	+2.37	-0.83
	0.6	+6.69	**+8.02**	-7.47	+3.45	**+6.70**	**-4.90**	+2.62	**+4.71**	-3.55	+0.58	-0.55	+0.03
	0.7	**+8.38**	+7.93	**-8.16**	**+3.65**	+5.26	-4.38	**+2.83**	+4.70	**-3.67**	**+2.46**	*-0.42*	-0.88
	0.8	+6.48	+6.27	-6.39	+2.18	+3.54	-2.80	+1.81	+3.81	-2.70	+1.67	+2.30	-2.01
	0.9	+4.02	+4.07	-4.07	+1.70	+3.10	-2.33	+0.62	+3.82	-2.03	+1.33	**+2.70**	**-2.06**

Table 3: Improvement on word alignment quality on small corpus after 8 iterations. BL stands for baseline, and NC represents unconstrained Model 4 training, and 0.9, 0.8, 0.7, 0.6 are the thresholds used in alignment link filtering.

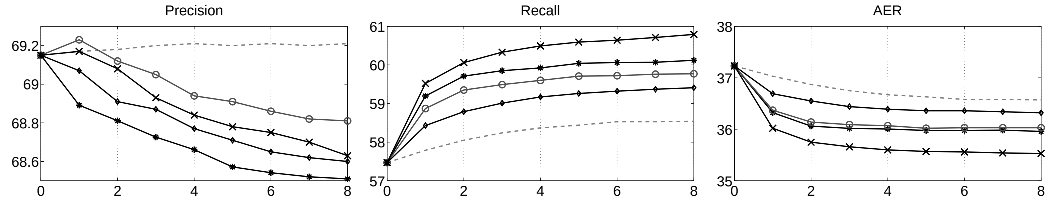
gorithm, the numbers are averaged across all iterations, the actual numbers of each iteration only have small differences. Although one might expect that the quality of resulting alignment from constrained EM be proportional to the quality of constraints, from the numbers in Table 2 and 3, we are not able to induce a clear relationship between them, and it could be language- or corpus-dependent. However, in practice we nonetheless use a held-out test set to tune this parameter. The



(a) Chinese-English



(b) English-Chinese



(c) Heuristically-symmetrized

Figure 4: Alignment qualities of each iteration for Chinese-English word alignment task. The grey dash curves represent unconstrained Model 4 training, and the curves with star, circle, cross and diamond markers are constrained EM alignments with 0.6, 0.7, 0.8 and 0.9 filtering thresholds respectively.

	Ch-En			**En-Ch**			**Heuristic**			**Discriminative**		
	P	R	AER	P	R	AER	P	R	AER	P	R	AER
BL	73.51	50.14	40.38	68.82	57.66	37.31	72.98	60.23	34.01	72.10	61.63	33.55
NC	73.23	50.38	40.30	68.30	58.00	37.27	72.39	60.99	33.80	72.07	61.81	33.45
0.8	76.27	52.90	37.53	70.26	60.26	35.11	72.75	63.49	32.19	72.64	63.29	32.35

Table 4: Improvement on word alignment quality on moderate-size corpus, where BL and NC represents baseline and non-constrained Model 4 training

relationship between quality of constraints and alignment results is an interesting topic for future research.

5.2 Effects on translation quality

In this experiment we run the whole machine translation pipeline and evaluate the system on BLEU score. We used the corpus LDC2006G05 which contains 25 million words as training set, the same discriminative tuning set as previously used (100 sentence pairs) and the remaining 21,763 sentence pairs from the hand-aligned corpus of the previous experiment are held-out test set for alignment qualities. A 4-gram language model trained from English GigaWord V1 and V2 corpus was used. The AER scores on the held-out test set are also provided for every iteration. Based on the observation in last experiment, we adopt the filtering threshold of 0.8.

Similar to previous experiment, the heuristically symmetrized alignments have lower precisions than their EMDC counterparts, however the gaps are smaller as shown in Table 4. We observe 2.85 and 2.21 absolute AER reduction on two directions, after symmetrization the gain on AER is 1.82. Continuing Model 4 training appears to have minimal effect on AER, and the improve-

I	M	NIST							GALE				
		mt06	**mt02**	**mt03**	**mt04**	**mt05**	**mt08**	*ain*	**db-nw**	**db-wb**	**dd-nw**	**dd-wb**	*aia*
0	G	31.00	31.80	29.89	32.63	29.33	24.24		26.92	24.48	28.44	24.26	
1	D	30.65	31.60	30.04	32.89	29.34	24.52	0.12	27.43	24.72	28.32	24.30	0.14
	G	31.35	31.91	30.35	32.75	29.40	24.16	0.15	27.39	24.50	28.22	24.60	0.15
2	D	**31.61**	32.31	30.40	33.06	29.49	24.11	0.33	**28.17**	24.42	28.58	24.36	0.34
	G	31.14	31.94	30.42	32.86	29.49	24.15	0.20	27.31	24.51	27.50	24.02	0.03
3	D	31.29	**32.39**	30.28	33.19	29.60	24.41	0.43	27.64	**25.32**	28.55	24.71	0.47
	G	30.94	31.95	30.15	32.71	29.38	24.22	0.12	27.63	24.61	28.80	25.05	0.29
4	D	30.80	32.04	30.51	33.24	29.49	24.61	0.46	27.61	25.27	28.72	24.98	0.53
	G	30.68	31.81	30.33	33.05	29.28	24.41	0.26	27.20	24.79	28.43	24.50	0.24
5	D	30.93	31.89	29.96	32.89	29.37	**24.50**	0.17	27.75	24.50	**29.05**	24.90	0.33
	G	31.16	32.28	**30.72**	**33.30**	**29.83**	24.30	**0.51**	27.32	25.05	28.60	**25.44**	**0.54**

Table 5: Improvement on translation alignment quality on moderate-size corpus, The column *ain* shows the average improvement of BLEU scores for all NIST test sets (excluding the tuning set MT06), and column *aia* is the average improvement on all unseen test sets. The column *M* indicates the alignment source, *G* means the alignment comes from generative aligner, and *D* means discriminative aligner respectively. The number of iterations is shown in column *I*.

ment mainly comes from the constraints.

In the experiment, we use the Moses toolkit to extract phrases, tune parameters and decode. We use the NIST MT06 test set as the tuning set, NIST MT02-05 and MT08 as unseen test sets. We also include results for four additional unseen test sets used in GALE evaluations: DEV07-Dev newswire part (dd-nw, 278 sentences) and Weblog part (dd-wb, 345 sentences), Dev07-Blind newswire part (db-nw, 276 sentences and Weblog part (db-wb, 312 sentences). Table 5 presents the average improvement on BLEU scores in each iteration. As we can see from the results, in all iterations we got improvement on BLEU scores, and the largest gain we have gotten is on the fifth iteration, which has 0.51 average improvement on five NIST test sets, and 0.54 average improvement across all nine test sets.

6 Conclusion

In this paper we presented a novel training scheme for word alignment task called EMDC. We also presented an extension of GIZA++ that can perform constrained EM training. By integrating it with a CRF-based discriminative word aligner and alignment link filtering, we can improve the alignment quality of both aligners iteratively. We experimented with small-size Chinese-English and Arabic English and moderate-size Chinese-English word alignment tasks, and observed in all four mono-directional alignments more than 3% absolute reduction on AER, with the largest improvement being 8.16% absolute on Arabic-to-English comparing to the baseline, and 5.90% comparing to Model 4 training with the same numbers of iterations. On a moderate-size Chinese-to-English tasks we also evaluated the impact of the improved alignment on translation quality across nine test sets. The 2% absolute AER reduction resulted in 0.5 average improvement on BLEU score.

Observations on the results raise several interesting questions for future research, such as 1) What is the relationship between the precision of the constraints and the quality of resulting alignments after iterations, 2) The effect of using different discriminative aligners, 3) Using aligners that explicitly model empty words and null alignments to provide additional constraints. We will continue exploration on these directions.

The extended GIZA++ is released to the research community as a branch of MGIZA++ (Gao and Vogel, 2008), which is available online[3].

Acknowledgement

This work is supported by NSF CluE Project (NSF 08-560) and DARPA GALE project.

[3]Accessible on Source Forge, with the URL: `http://sourceforge.net/projects/mgizapp/`

References

Ayan, Necip Fazil and Bonnie J. Dorr. 2006. Going beyond aer: an extensive analysis of word alignments and their impact on mt. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 9–16.

Blunsom, Phil and Trevor Cohn. 2006. Discriminative word alignment with conditional random fields. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 65–72.

Brown, Peter F., Vincent J.Della Pietra, Stephen A. Della Pietra, and Robert. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. In *Computational Linguistics*, volume 19(2), pages 263–331.

Callison-Burch, C., D. Talbot, and M. Osborne. 2004. Statistical machine translation with word- and sentence-aligned parallel corpora. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, pages 175–183.

Fraser, Alexander and Daniel Marcu. 2006. Semi-supervised training for statistical word alignment. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 769–776.

Fraser, Alexander and Daniel Marcu. 2007. Getting the structure right for word alignment: LEAF. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 51–60.

Gao, Qin and Stephan Vogel. 2008. Parallel implementations of word alignment tool. In *Proceedings of the ACL 2008 Software Engineering, Testing, and Quality Assurance Workshop*, pages 49–57.

Gao, Qin and Stephan Vogel. 2010. Consensus versus expertise : A case study of word alignment with mechanical turk. In *NAACL 2010 Workshop on Creating Speech and Language Data With Mechanical Turk*, pages 30–34.

Gao, Qin, Nguyen Bach, and Stephan Vogel. 2010. A semi-supervised word alignment algorithm with partial manual alignments. In *In Proceedings of the ACL 2010 joint Fifth Workshop on Statistical Machine Translation and Metrics MATR (ACL-2010 WMT)*.

Guzman, Francisco, Qin Gao, and Stephan Vogel. 2009. Reassessment of the role of phrase extraction in pbsmt. In *The twelfth Machine Translation Summit*.

Huang, Fei. 2009. Confidence measure for word alignment. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 932–940.

Ittycheriah, Abraham and Salim Roukos. 2005. A maximum entropy word aligner for arabic-english machine translation. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 89–96.

Liu, Yang, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 459–466.

Moore, Robert C. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 81–88.

Niehues, Jan. and Stephan. Vogel. 2008. Discriminative word alignment via alignment matrix modeling. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 18–25.

Och, Franz Joseph and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. In *Computational Linguistics*, volume 1:29, pages 19–51.

Taskar, Ben, Simon Lacoste-Julien, and Dan Klein. 2005. A discriminative matching approach to word alignment. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 73–80.

Vogel, Stephan, Hermann Ney, and Christoph Tillmann. 1996. HMM based word alignment in statistical machine translation. In *Proceedings of 16th International Conference on Computational Linguistics)*, pages 836–841.

A Large Scale Ranker-Based System for Search Query Spelling Correction

Jianfeng Gao
Microsoft Research, Redmond
jfgao@microsoft.com

Xiaolong Li
Microsoft Corporation
xiaolong.li@microsoft.com

Daniel Micol
Microsoft Corporation
danielmi@microsoft.com

Chris Quirk
Microsoft Research, Redmond
chrisq@microsoft.com

Xu Sun
University of Tokyo
xusun@mist.i.u-tokyo.ac.jp

Abstract

This paper makes three significant extensions to a noisy channel speller designed for standard written text to target the challenging domain of search queries. First, the noisy channel model is subsumed by a more general ranker, which allows a variety of features to be easily incorporated. Second, a distributed infrastructure is proposed for training and applying Web scale *n*-gram language models. Third, a new phrase-based error model is presented. This model places a probability distribution over transformations between multi-word phrases, and is estimated using large amounts of query-correction pairs derived from search logs. Experiments show that each of these extensions leads to significant improvements over the state-of-the-art baseline methods.

1 Introduction

Search queries present a particular challenge for traditional spelling correction methods. New search queries emerge constantly. As a result, many queries contain valid search terms, such as proper nouns and names, which are not well established in the language. Therefore, recent research has focused on the use of Web corpora and search logs, rather than human-compiled lexicons, to infer knowledge about spellings and word usages in search queries (e.g., Whitelaw et al., 2009; Cucerzan and Brill, 2004).

The spelling correction problem is typically formulated under the framework of the noisy channel model. Given an input query $Q = q_1 ... q_I$, we want to find the best spelling correction $C = c_1 ... c_J$ among all candidates:

$$C^* = \underset{C}{\operatorname{argmax}} P(C|Q) \quad (1)$$

Applying Bayes' Rule, we have

$$C^* = \underset{C}{\operatorname{argmax}} P(Q|C)P(C) \quad (2)$$

where the error model $P(Q|C)$ models the transformation probability from C to Q, and the language model (LM) $P(C)$ models the likelihood that C is a correctly spelled query.

This paper extends a noisy channel speller designed for regular text to search queries in three ways: using a ranker (Section 3), using Web scale LMs (Section 4), and using phrase-based error models (Section 5).

First of all, we propose a ranker-based speller that covers the noisy channel model as a special case. Given an input query, the system first generates a short list of candidate corrections using the noisy channel model. Then a feature vector is computed for each query and candidate correction pair. Finally, a ranker maps the feature vector to a real-valued score, indicating the likelihood that this candidate is a desirable correction. We will demonstrate that ranking provides a flexible modeling framework for incorporating a wide variety of features that would be difficult to model under the noisy channel framework.

Second, we explore the use of Web scale LMs for query spelling correction. While traditional LM research focuses on how to make the model "smarter" via how to better estimate the probability of unseen words (Chen and Goodman, 1999); and how to model the grammatical structure of language (e.g., Charniak, 2001), recent studies show that significant improvements can be achieved using "stupid" *n*-gram models trained on very large corpora (e.g., Brants et al., 2007). We adopt the latter strategy in this study. We present a distributed infrastructure to efficiently train and apply Web scale LMs. In addition, we observe that search queries are composed in a language style different from that of regular text. We thus train multiple LMs using different texts associated with Web corpora and search queries.

Third, we propose a phrase-based error model that captures the probability of transforming one

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 358–366,
Beijing, August 2010

multi-term phrase into another multi-term phrase. Compared to traditional error models that account for transformation probabilities between single characters or substrings (e.g., Kernighan et al., 1990; Brill and Moore, 2000), the phrase-based error model is more effective in that it captures inter-term dependencies crucial for correcting real-word errors, prevalent in search queries. We also present a novel method of extracting large amounts of query-correction pairs from search logs. These pairs, implicitly judged by millions of users, are used for training the error models.

Experiments show that each of the extensions leads to significant improvements over its baseline methods that were state-of-the-art until this work, and that the combined method yields a system which outperforms the noisy channel speller by a large margin: a 6.3% increase in accuracy on a human-labeled query set.

2 Related Work

Prior research on spelling correction for regular text can be grouped into two categories: correcting non-word errors and real-word errors. The former focuses on the development of error models based on different edit distance functions (e.g., Kucich, 1992; Kernighan et al., 1990; Brill and Moore, 2000; Toutanova and Moore, 2002). Brill and Moore's substring-based error model, considered to be state-of-the-art among these models, acts as the baseline against which we compare our models. On the other hand, real-word spelling correction tries to detect incorrect usages of a valid word based on its context, such as "peace" and "piece" in the context "a _ of cake". N-gram LMs and naïve Bayes classifiers are commonly used models (e.g., Golding and Roth, 1996; Mangu and Brill, 1997; Church et al., 2007).

While almost all of the spellers mentioned above are based on a pre-defined dictionary (either a lexicon against which the edit distance is computed, or a set of real-word confusion pairs), recent research on query spelling correction focuses on exploiting noisy Web corpora and query logs to infer knowledge about spellings and word usag in queries (Cucerzan and Brill 2004; Ahmad and Kondrak, 2005; Li et al., 2006; Whitelaw et al., 2009). Like those spellers designed for regular text, most of these query spelling systems are also based on the noisy channel framework.

3 A Ranker-Based Speller

The noisy channel model of Equation (2) does not have the flexibility to incorporate a wide variety of features useful for spelling correction, e.g., whether a candidate appears as a Wikipedia document title. We thus generalize the speller to a ranker-based system. Let **f** be a feature vector of a query and candidate correction pair (Q, C). The ranker maps **f** to a real value y that indicates how likely C is a desired correction. For example, a linear ranker maps **f** to y with a weight vector **w** such as $y = \mathbf{w} \cdot \mathbf{f}$, where **w** is optimized for accuracy on human-labeled (Q, C) pairs. Since the logarithms of the LM and error model probabilities can be included as features, the ranker covers the noisy channel model as a special case.

For efficiency, our speller operates in two distinct stages: candidate generation and re-ranking.

In candidate generation, an input query is first tokenized into a sequence of terms. For each term q, we consult a lexicon to identify a list of spelling suggestions c whose edit distance from q is lower than some threshold. Our lexicon contains around 430,000 high frequency query unigram and bigrams collected from 1 year of query logs. These suggestions are stored in a lattice.

We then use a decoder to identify the 20-best candidates from the lattice according to Equation (2), where the LM is a backoff bigram model trained on 1 year of query logs, and the error model is approximated by weighted edit distance:

$$-\log P(Q|C) \propto \mathrm{EditDist}(Q, C) \quad (3)$$

The decoder uses a standard two-pass algorithm. The first pass uses the Viterbi algorithm to find the best C according to the model of Equations (2) and (3). The second pass uses the A-star algorithm to find the 20-best corrections, using the Viterbi scores computed at each state in the first pass as heuristics.

The core component in the second stage is a ranker, which re-ranks the 20-best candidate corrections using a set of features extracted from (Q, C). If the top C after re-ranking is different from Q, C is proposed as the correction. We use 96 features in this study. In addition to the two features derived from the noisy channel model, the rest of the features can be grouped into the following 5 categories.

1. **Surface-form similarity features**, which check whether C and Q differ in certain patterns,

e.g., whether C is transformed from Q by adding an apostrophe, or by adding a stop word at the beginning or end of Q.

2. **Phonetic-form similarity features**, which check whether the edit distance between the metaphones (Philips, 1990) of a query term and its correction candidate is below some thresholds.

3. **Entity features**, which check whether the original query is likely to be a proper noun based on an in-house named entity recognizer.

4. **Dictionary features**, which check whether a query term or a candidate correction are in one or more human-compiled dictionaries, such as the extracted Wiki, MSDN, and ODP dictionaries.

5. **Frequency features**, which check whether the frequency of a query term or a candidate correction is above certain thresholds in different datasets, such as query logs and Web documents.

4 Web Scale Language Models

An n-gram LM assigns a probability to a word string $w_1^L = (w_1, \ldots, w_L)$ according to

$$P(w_1^L) = \prod_{i=1}^{L} P(w_i | w_1^{i-1}) \approx \prod_{i=1}^{L} P(w_i | w_{i-n+1}^{i-1}) \quad (4)$$

where the approximation is based on a Markov assumption that each word depends only upon the immediately preceding n-1 words. In a speller, the log of n-gram LM probabilities of an original query and its candidate corrections are used as features in the ranker.

While recent research reports the benefits of large LMs trained on Web corpora on a variety of applications (e.g. Zhang et al., 2006; Brants et al., 2007), it is also clear that search queries are composed in a language style different from that of the body or title of a Web document. Thus, in this study we developed a set of large LMs from different text streams of Web documents and query logs. Below, we first describe the n-gram LM collection used in this study, and then present a distributed n-gram LM platform based on which these LMs are built and served for the speller.

4.1 Web Scale Language Models

Table 1 summarizes the data sets and Web scale n-gram LMs used in this study. The collection is built from high quality English Web documents containing trillions of tokens, served by a popular commercial search engine. The collection consists of several data sets built from different Web sources, including the different text fields from the Web documents (i.e., body, title, and anchor texts) and search query logs. The raw texts extracted from these different sources were pre-processed in the following manner: texts are tokenized based on white-space and upper case letters are converted to lower case. Numbers are retained, and no stemming/inflection is performed. The n-gram LMs are word-based backoff models, where the n-gram probabilities are estimated using Maximum Likelihood Estimation with smoothing. Specifically, for a trigram model, the smoothed probability is computed as

Dataset	**Body**	**Anchor**	**Title**	**Query**
Total tokens	1.3T	11.0B	257.2B	28.1B
Unigrams	1.2B	60.3M	150M	251.5M
Bigrams	11.7B	464.1M	1.1B	1.3B
Trigrams	60.0B	1.4B	3.1B	3.1B
4-grams	148.5B	2.3B	5.1B	4.6B
Size on disk[#]	12.8TB	183GB	395GB	393GB

[#] N-gram entries as well as other model parameters are stored.

Table 1: Statistics of the Web n-gram LMs collection (count cutoff = 0 for all models). These models will be accessible at Microsoft (2010).

$$P(w_i | w_{i-2} w_{i-1}) = \quad (5)$$
$$\begin{cases} \dfrac{C(w_{i-2} w_{i-1} w_i) - D(C(w_{i-2} w_{i-1} w_i))}{C(w_{i-2} w_{i-1})} & \text{if } C(w_{i-2} w_{i-1} w_i) \\ \alpha(w_{i-2} w_{i-1}) P(w_i | w_{i-1}) & \text{otherwise} \end{cases}$$

where $C(\cdot)$ is the count of the n-gram in the training corpus and α is a normalization factor. $D(C)$ is a discount function for smoothing. We use modified absolute discounting (Gao et al., 2001), whose parameters can be efficiently estimated and performance converges to that of more elaborate state-of-the-art techniques like Kneser-Ney smoothing in large data (Nguyen et al. 2007).

4.2 Distributed N-gram LM Platform

The platform is developed on a distributed computing system designed for storing and analyzing massive data sets, running on large clusters consisting of hundreds of commodity servers connected via high-bandwidth network.

We use the SCOPE (Structured Computations Optimized for Parallel Execution) programming model (Chaiken et al., 2008) to train the Web scale n-gram LMs shown in Table 1. The SCOPE scripting language resembles SQL which many programmers are familiar with. It also supports

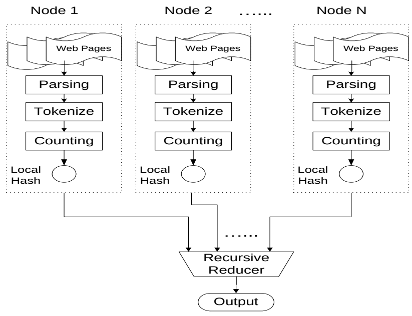


Figure 1. Distributed 5-gram counting.

C# expressions so that users can easily plug-in customized C# classes. SCOPE supports writing a program using a series of simple data transformations so that users can simply write a script to process data in a *serial* manner without wondering how to achieve parallelism while the SCOPE compiler and optimizer are responsible for translating the script into an efficient, parallel execution plan. We illustrate the usage of SCOPE for building LMs using the following example of counting 5-grams from the body text of English Web pages. The flowchart is shown in Figure 1.

The program is written in SCOPE as a step-by- step of computation, where a command takes the output of the previous command as its input.

```
ParsedDoc=SELECT docId, TokenizedDoc
   FROM @"/shares/…/EN_Body.txt"
   USING DefaultTextExtractor;

NGram=PROCESS ParsedDoc
   PRODUCE NGram, NGcount
   USING NGramCountProcessor(-stream
   TokenizedDoc -order 5 –bufferSize
   20000000);

NGramCount=REDUCE NGram
   ON NGram
   PRODUCE NGram, NGcount
   USING NGramCountReducer;

OUTPUT TO @"Body-5-gram-count.txt";
```

The first SCOPE command is a SELECT statement that extracts parsed Wed body text. The second command uses a build-in Processor (NGramCountProcessor) to map the parsed documents into separate *n*-grams together with their counts. It generates a local hash at each node (i.e., a core in a multi-core server) to store the (*n*-gram, count) pairs. The third command (REDUCE) aggregates counts from different nodes according to the key (*n*-gram string). The final command (OUTPUT) writes out the resulting to a data file.

C:	"disney theme park"	*correct query*
S:	["disney", "theme park"]	*segmentation*
T:	["disnee", "theme part"]	*translation*
M:	(1 → 2, 2→ 1)	*permutation*
Q:	"theme part disnee"	*misspelled query*

Figure 2: Example demonstrating the generative procedure behind the phrase-based error model.

The smoothing method can be implemented similarly by the customized smoothing Processor/Reducer. They can be imported from the existing C# codes (e.g., developed for building LMs in a single machine) with minor changes.

It is straightforward to apply the built LMs for the ranker in the speller. The *n*-gram platform provides a DLL for *n*-gram batch lookup. In the server, an *n*-gram LM is stored in the form of multiple lists of key-value pairs, where the key is the hash of an *n*-gram string and the value is either the *n*-gram probability or backoff parameter.

5 Phrase-Based Error Models

The goal of an error model is to transform a correctly spelled query *C* into a misspelled query *Q*. Rather than replacing single words in isolation, the phrase-based error model replaces sequences of words with sequences of words, thus incorporating contextual information. The training procedure closely follows Sun et al. (2010). For instance, we might learn that "*theme part*" can be replaced by "*theme park*" with relatively high probability, even though "*part*" is not a misspelled word. We use this generative story: first the correctly spelled query *C* is broken into *K* non-empty word sequences $\mathbf{c}_1$, …, $\mathbf{c}_k$, then each is replaced with a new non-empty word sequence $\mathbf{q}_1$, …, $\mathbf{q}_k$, finally these phrases are permuted and concatenated to form the misspelled *Q*. Here, **c** and **q** denote consecutive sequences of words.

To formalize this generative process, let *S* denote the segmentation of *C* into *K* phrases $\mathbf{c}_{1…}\mathbf{c}_K$, and let *T* denote the *K* replacement phrases $\mathbf{q}_1…\mathbf{q}_K$ – we refer to these ($\mathbf{c}_i$, $\mathbf{q}_i$) pairs as *bi-phrases*. Finally, let *M* denote a permutation of *K* elements representing the final reordering step. Figure 2 demonstrates the generative procedure.

Next let us place a probability distribution over rewrite pairs. Let *B*(*C*, *Q*) denote the set of *S*, *T*, *M* triples that transform *C* into *Q*. Assuming a uniform probability over segmentations, the phrase-based probability can be defined as:

$$P(Q|C) \propto \sum_{(S,T,M)\in B(C,Q)} P(T|C,S) \cdot P(M|C,S,T) \quad (6)$$

As is common practice in SMT, we use the maximum approximation to the sum:

$$P(Q|C) \approx \max_{(S,T,M)\in B(C,Q)} P(T|C,S) \cdot P(M|C,S,T) \quad (7)$$

5.1 Forced Alignments

Although we have defined a generative model for transforming queries, our goal is not to propose new queries, but rather to provide scores over existing *Q* and *C* pairs that will act as features for the ranker. Furthermore, the word-level alignments between *Q* and *C* can most often be identified with little ambiguity. Thus we restrict our attention to those phrase transformations consistent with a good word-level alignment.

Let *J* be the length of *Q*, *L* be the length of *C*, and $A = a_1 \ldots a_J$ be a hidden variable representing the word alignment between them. Each a_i takes on a value ranging from 1 to *L* indicating its corresponding word position in *C*, or 0 if the *i*th word in *Q* is unaligned. The cost of assigning *k* to a_i is equal to the Levenshtein edit distance (Levenshtein, 1966) between the i^{th} word in *Q* and the k^{th} word in *C*, and the cost of assigning 0 to a_i is equal to the length of the i^{th} word in *Q*. The least cost alignment A^* between *Q* and *C* is computed efficiently using the A-star algorithm.

When scoring a given candidate pair, we further restrict our attention to those *S*, *T*, *M* triples that are consistent with the word alignment, which we denote as $B(C, Q, A^*)$. Here, consistency requires that if two words are aligned in A^*, then they must appear in the same bi-phrase ($\mathbf{c}_i$, $\mathbf{q}_i$). Once the word alignment is fixed, the final permutation is uniquely determined, so we can safely discard that factor. Thus we have:

$$P(Q|C) \approx \max_{\substack{(S,T,M)\in \\ \mathcal{B}(C,Q,A^*)}} P(T|C,S) \quad (8)$$

For the sole remaining factor $P(T|C, S)$, we make the assumption that a segmented query $T = \mathbf{q}_1 \ldots \mathbf{q}_K$ is generated from left to right by transforming each phrase $\mathbf{c}_1 \ldots \mathbf{c}_K$ independently:

$$P(T|C,S) = \prod_{k=1}^{K} P(\mathbf{q}_k|\mathbf{c}_k), \quad (9)$$

where $P(\mathbf{q}_k|\mathbf{c}_k)$ is a phrase transformation probability, the estimation of which will be described in Section 5.2.

Google:

```
http://www.google.com/search?
hl=en&source=hp&
q=harrypotter+sheme+part&aq=f&oq=&aqi=

http://www.google.com/search?
hl=en&ei=rnNAS8-oKsWe_AaB2eHlCA&
sa=X&oi=spell&resnum=0&ct=
result&cd=1&ved=0CA4QBSgA&
q=harry+potter+theme+park&spell=1
```

Yahoo:

```
http://search.yahoo.com/search;
_ylt=A0geu6ywckBL_XIBSDtXNyoA?
p=harrypotter+sheme+part&
fr2=sb-top&fr=yfp-t-701&sao=1

http://search.yahoo.com/search?
ei=UTF-8&fr=yfp-t-701&
p=harry+potter+theme+park
&SpellState=n-2672070758_q-tsI55N6srhZa.
qORA0MuawAAAA%40%40&fr2=sp-top
```

Bing:

```
http://www.bing.com/search?
q=harrypotter+sheme+part&form=QBRE&qs=n

http://www.bing.com/search?
q=harry+potter+theme+park&FORM=SSRE
```

Figure 3. A sample of query reformulation sessions from 3 popular search engines. These sessions show that a user first issues the query "harrypotter sheme part", and then clicks on the resulting spell suggestion "harry potter theme park".

To find the maximum probability assignment efficiently, we use a dynamic programming approach, similar to the monotone decoding algorithm described in Och (2002).

5.2 Training the Error Model

Given a set of (*Q*, *C*) pairs as training data, we follow a method commonly used in SMT (Och and Ney, 2004) to extract bi- phrases and estimate their replacement probabilities. A detailed description is discussed in Sun et al. (2010).

We now describe how (*Q*, *C*) pairs are generated automatically from massive *query reformulation sessions* of a commercial Web browser.

A query reformulation session contains a list of URLs that record user behaviors that relate to the query reformulation functions, provided by a Web search engine. For example, most commercial search engines offer the "did you mean" function, suggesting a possible alternate interpretation or spelling of a user-issued query. Figure 3 shows a sample of the query reformulation sessions that record the "did you mean" sessions from three of the most popular search engines. These sessions encode the same user behavior: A user first queries for "harrypotter sheme part", and then clicks on the resulting spelling suggestion "harry potter theme park". We can "reverse-engineer" the parameters from the URLs of these sessions, and deduce how each search engine encodes both a query and the fact that a user arrived at a URL by clicking on the spelling suggestion of the query – an strong indication that the spelling suggestion is desired. In this study, from 1 year of sessions, we extracted ~120 million pairs. We found the data set very clean because these spelling corrections are actually clicked, and thus judged implicitly, by many users.

In addition to the "did you mean" functionality, recently some search engines have introduced two new spelling suggestion functions. One is the "auto-correction" function, where the search engine is confident enough to automatically apply the spelling correction to the query and execute it to produce search results. The other is the "split pane" result page, where one half portion of the search results are produced using the original query, while the other half, usually visually separate portion of results, are produced using the auto-corrected query.

In neither of these functions does the user ever receive an opportunity to approve or disapprove of the correction. Since our extraction approach focuses on user-approved spelling suggestions, we ignore the query reformulation sessions recording either of the two functions. Although by doing so we could miss some basic, obvious spelling corrections, our experiments show that the negative impact on error model training is negligible. One possible reason is that our baseline system, which does not use any error model learned from the session data, is already able to correct these basic, obvious spelling mistakes. Thus, including these data for training is unlikely to bring any further improvement.

We found that the error models trained using the data directly extracted from the query reformulation sessions suffer from the problem of underestimating the self-transformation probability of a query $P(Q_2{=}Q_1|Q_1)$, because we only included in the training data the pairs where the query is different from the correction. To deal with this problem, we augmented the training data by including correctly spelled queries, i.e., the pairs (Q_1, Q_2) where $Q_1 = Q_2$. First, we extracted a set of queries from the sessions where no spell suggestion is presented or clicked on. Second, we removed from the set those queries that were recognized as being auto-corrected by a search engine. We do so by running a sanity check of the queries against our baseline noisy channel speller, which will be described in Section 6. If the system consider a query misspelled, we assumed it an obvious misspelling, and removed it. The remaining queries were assumed to be correctly spelled and were added to the training data.

6 Experiments

We perform the evaluation using a manually annotated data set containing 24,172 queries sampled from one year's query logs from a commercial search engine. The spelling of each query is manually corrected by four independent annotators. The average length of queries in the data sets is 2.7 words. We divided the data set into non-overlapped training and test data sets. The training data contain 8,515 (Q, C) pairs, among which 1,743 queries are misspelled (i.e. $Q \neq C$). The test data contain 15,657 (Q, C) pairs, among which 2,960 queries are misspelled.

The speller systems we developed in this study are evaluated using the following metrics.

- **Accuracy:** The number of correct outputs generated by the system divided by the total number of queries in the test set.
- **Precision:** The number of correct spelling corrections for misspelled queries generated by the system divided by the total number of corrections generated by the system.
- **Recall:** The number of correct spelling corrections for misspelled queries generated by the system divided by the total number of misspelled queries in the test set.

We also perform a significance test, a t-test with a significance level of 0.05.

In our experiments, all the speller systems are ranker-based. Unless otherwise stated, the ranker is a two-layer neural net with 5 hidden nodes. The free parameters of the neural net are trained to optimize accuracy on the training data using the back propagation algorithm (Burges et al., 2005) .

6.1 System Results

Table 1 summarizes the main results of different spelling systems. Row 1 is the baseline speller where the noisy channel model of Equations (2)

#	System	Accuracy	Precision	Recall
1	Noisy channel	85.3	72.1	35.9
2	Linear ranker	88.0	74.0	42.8
3	Nonlinear ranker	89.0	74.1	49.6
4	3 + PBEM	90.7	78.7	58.2
5	3 + WLMs	90.4	75.1	58.7
6	3 + PBEM + WLMs	91.6	79.1	63.9

Table 1. Summary of spelling correction results.

and (3) is used. The error model is based on the weighted edit distance function and the LM is a backoff bigram model trained on 1 year of query logs, with count cutoff 30. Row 2 is the speller using a linear ranker to incorporate all ranking features described in Section 3. The weights of the linear ranker are optimized using the Averaged Perceptron algorithm (Freund and Schapire, 1999). Row 3 is the speller where a nonlinear ranker (i.e., 2-layer neural net) is trained atop the features. Rows 4, 5 and 6 are systems that incorporate the additional features derived from the phrase-based error model (PBEM) described in Section 5 and the four Web scale LMs (WLMs) listed in Table 1.

The results show that (1) the ranker is a very flexible modeling framework where a variety of fine-grained features can be easily incorporated, and a ranker-based speller outperforms significantly ($p < 0.01$) the traditional system based on the noisy channel model (Row 2 vs. Row 1); (2) the speller accuracy can be further improved by using more sophisticated rankers and learning algorithms (Row 3 vs. Row 2); (3) both WLMs and PBEM bring significant improvements (Rows 4 and 5 vs. Row 3); and (4) interestingly, the gains from WLMs and PBEM are additive and the combined leads to a significantly better speller (Row 6 vs. Rows 4 and 5) than that of using either of them individually.

In what follows, we investigate in detail how the WLMs and PBEM trained on massive Web content and search logs improve the accuracy of the speller system. We will compare our models with the state-of-the-art models proposed previously. From now on, the system listed in Row 3 of Table 1 will be used as baseline.

6.2 Language Models

The quality of n-gram LMs depends on the order of the model, the size of the training data, and how well the training data match the test data. Figure 4 illustrates the perplexity results of the four LMs trained on different data sources tested on a random sample of 733,147 queries. The results show that (1) higher order LMs produce lower perplexities, especially when moving beyond unigram models; (2) as expected, the query LMs are most predictive for the test queries, though they are from independent query log snapshots; (3) although the body LMs are trained on much larger amounts of data than the title and anchor LMs, the former lead to much higher perplexity values, indicating that both title and anchor texts are quantitatively much more similar to queries than body texts.

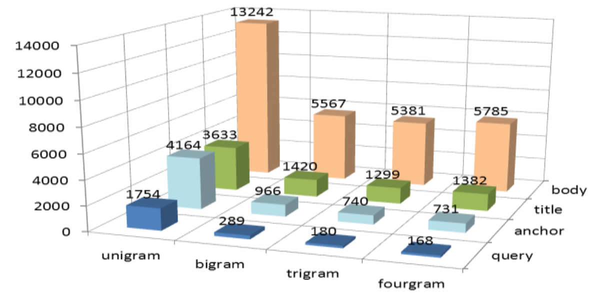


Figure 4. Perplexity results on test queries, using n-gram LMs with different orders, derived from different data sources.

Table 2 summarizes the spelling results using different LMs. For comparison, we also built a 4-gram LM using the Google 1T web 5-gram corpus (Brants and Franz, 2006). This model is referred to as the G1T model, and is trained using the “stupid backoff” smoothing method (Brants et al., 2007). Due to the high count cutoff applied by the Google corpus (i.e., n-grams must appear at least 40 times to be included in the corpus), we found the G1T model results to a higher OOV rate (i.e., 6.5%) on our test data than that of the 4 Web scale LMs (i.e., less than 1%).

The results in Table 2 are more or less consistent with the perplexity results: the query LM is the best performer; there is no significant difference among the body, title and anchor LMs though the body LM is trained on a much larger amount of data; and all the 4 Web scale LMs outperform the G1T model substantially due to the significantly lower OOV rates.

6.3 Error Models

This section compares the phrase-based error model (PBEM) described in Section 5, with one of the state-of-the-art error models, proposed by Brill and Moore (2000), henceforth referred to as

#	System	Accuracy	Precision	Recall
1	Baseline	89.0	74.1	49.6
2	1+ query 4-gram	90.1	75.6	56.3
3	1 + body 4-gram	89.9	75.7	54.4
4	1 + title 4-gram	89.8	75.4	54.7
5	1 + anchor 4-gram	89.9	75.1	55.6
6	1 + G1T 4-gram	89.4	75.1	51.5

Table 2. Spelling correction results using different LMs trained on different data sources.

#	System	Accuracy	Precision	Recall
1	Baseline w/o EM	88.6	72.0	47.0
2	Baseline	89.0	74.1	49.6
3	1 + B&M, *L*=1	89.0	73.3	50.1
4	1 + B&M, *L*=3	89.2	73.7	51.0
5	1 + PBEM, *L*=1	90.1	76.7	55.6
6	1 + PBEM, *L*=3	90.7	78.5	58.1

Table 3. Spelling correction results using different error models.

the B&M model. B&M is a substring error model. It estimates $P(q|c)$ as

$$P(q|c) \approx \max_{\substack{R,T \\ s.t. |T|=|R|}} \prod_{i=1}^{|R|} P(T_i|R_i), \quad (10)$$

where R is a partitioning of correction term c into adjacent substrings, and T is a partitioning of query term q, such that $|T|=|R|$. The partitions are thus in one-to-one alignment. To train the B&M model, we extracted 1 billion term-correction pairs (q, c) from the set of 120 million query-correction pairs (Q, C), derived from the search logs as described in Section 5.2.

Table 3 summarizes the comparison results. Rows 1 and 2 are our ranker-based baseline systems with and without the error model (EM) feature. The error model is based on weighted edit distance of Eq. (3), where the weights are learned on some manually annotated word-correction pairs (which is not used in this study). Rows 3 and 4 are the B&M models using different maximum substring lengths, specified by L. L=1 reduces B&M to the weighted edit distance model in Row 2. Rows 5 and 6 are PBEMs with different maximum phrase lengths. L=1 reduces PBEM to a word-based error model. The results show the benefits of capturing context information in error models. In particular, the significant improvements resulting from PBEM demonstrate that the dependencies between words are far more effective than that between characters (within a word) for spelling correction. This is largely due to the fact that there are many real-word spelling errors in search queries. We also notice that PBEM is a more powerful model than

#	# of word pairs	Accuracy	Precision	Recall
1	Baseline w/o EM	88.55	71.95	46.97
2	1M	89.15	73.71	50.74
3	10M	89.22	74.11	50.92
4	100M	89.20	73.60	51.06
5	1B	89.21	73.72	50.99

Table 4. The performance of B&M error model (*L*=3) as a function of the size of training data (# of word pairs).

#	# of (*Q*, *C*) pairs	Accuracy	Precision	Recall
1	Baseline w/o EM	88.55	71.95	46.97
2	5M	89.59	77.01	52.34
3	15M	90.23	77.87	56.67
4	45M	90.45	78.56	57.02
5	120M	90.70	78.49	58.12

Table 5. The performance of PBEM (*L*=3) as a function of the size of training data (# of (*Q*, *C*) pairs).

B&M in that it can benefit more from increasingly larger training data. As shown in Tables 4 and 5, whilst the performance of B&M saturates quickly with the increase of training data, the performance of PBEM does not appear to have peaked – further improvements are likely given a larger data set.

7 Conclusions and Future Work

This paper explores the use of massive Web corpora and search logs for improving a ranker-based search query speller. We show significant improvements over a noisy channel speller using fine-grained features, Web scale LMs, and a phrase-based error model that captures internword dependencies. There are several techniques we are exploring to make further improvements. First, since a query speller is developed for improving the Web search results, it is natural to use features from search results in ranking, as studied in Chen et al. (2007). The challenge is efficiency. Second, in addition to query reformulation sessions, we are exploring other search logs from which we might extract more (Q, C) pairs for error model training. One promising data source is clickthrough data (e.g., Agichtein et al, 2006; Gao et al., 2009). For instance, we might try to learn a transformation from the title or anchor text of a document to the query that led to a click on that document. Finally, the phrase-based error model is inspired by phrase-based SMT systems. We are introducing more SMT techniques such as alignment and translation rule exaction. In a broad sense, spelling correction can be viewed as a monolingual MT problem where we translate bad English queries into good ones.

Acknowledgments

The authors would like to thank Andreas Bode, Mei Li, Chenyu Yan and Kuansan Wang for the very helpful discussions and collaboration. The work was done when Xu Sun was visiting Microsoft Research Redmond.

References

Agichtein, E., Brill, E. and Dumais, S. 2006. Improving web search ranking by incorporating user behavior information. In *SIGIR*, pp. 19-26.

Ahmad, F., and Kondrak, G. 2005. Learning a spelling error model from search query logs. In *HLT-EMNLP*, pp. 955-962.

Brants, T., and Franz, A. 2006. Web 1T 5-gram corpus version 1.1. Technical report, Google Research.

Brants, T., Popat, A. C., Xu, P., Och, F. J., and Dean, J. 2007. Large language models in machine translation. In *EMNLP-CoNLL*, pp. 858 - 867.

Brill, E., and Moore, R. C. 2000. An improved error model for noisy channel spelling correction. In *ACL*, pp. 286-293.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, and Hullender, G. 2005. Learning to rank using gradient descent. In *ICML*, pp. 89-96.

Chaiken, R., Jenkins, B., Larson, P., Ramsey, B., Shakib, D., Weaver, S., and Zhou, J. 2008. SCOPE: easy and efficient parallel processing f massive data sets. In *Proceedings of the VLDB Endowment*, pp. 1265-1276.

Charniak, E. 2001. Immediate-head parsing for language models. In *ACL/EACL*, pp. 124-131.

Chen, S. F., and Goodman, J. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(10):359-394.

Chen, Q., Li, M., and Zhou, M. 2007. Improving query spelling correction using web search results. In *EMNLP-CoNLL*, pp. 181-189.

Church, K., Hard, T., and Gao, J. 2007. Compressing trigram language models with Golomb coding. In *EMNLP-CoNLL*, pp. 199-207.

Cucerzan, S., and Brill, E. 2004. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *EMNLP*, pp. 293-300.

Freund, Y. and Schapire, R. E. 1999. Large margin classification using the perceptron algorithm. In *Machine Learning*, 37(3): 277-296.

Gao, J., Goodman, J., and Miao, J. 2001. The use of clustering techniques for language modeling - application to Asian languages. *Computational Linguistics and Chinese Language Processing*, 6(1):27–60, 2001.

Gao, J., Yuan, W., Li, X., Deng, K., and Nie, J-Y. 2009. Smoothing clickthrough data for web search ranking. In *SIGIR*, pp. 355-362.

Golding, A. R., and Roth, D. 1996. Applying winnow to context-sensitive spelling correction. In *ICML*, pp. 182-190.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *SIGKDD*, pp. 133-142.

Kernighan, M. D., Church, K. W., and Gale, W. A. 1990. A spelling correction program based on a noisy channel model. In *COLING*, pp. 205-210.

Koehn, P., Och, F., and Marcu, D. 2003. Statistical phrase-based translation. In *HLT/NAACL*, pp. 127-133.

Kucich, K. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24(4):377-439.

Levenshtein, V. I. 1966. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707-710.

Li, M., Zhu, M., Zhang, Y., and Zhou, M. 2006. Exploring distributional similarity based models for query spelling correction. In *ACL*, pp. 1025-1032.

Mangu, L., and Brill, E. 1997. Automatic rule acquisition for spelling correction. In *ICML*, pp. 187-194.

Microsoft Microsoft web n-gram services. 2010. http://research.microsoft.com/web-ngram

Nguyen, P., Gao, J., and Mahajan, M. 2007. MSRLM: a scalable language modeling toolkit. Technical report TR-2007-144, Microsoft Research.

Och, F. 2002. *Statistical machine translation: from single-word models to alignment templates.* PhD thesis, RWTH Aachen.

Och, F., and Ney, H. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4): 417-449.

Philips, L. 1990. Hanging on the metaphone. *Computer Language Magazine*, 7(12):38-44.

Sun, X., Gao, J., Micol, D., and Quirk, C. 2010. Learning phrase-based spelling error models from clickthrough data. In *ACL*.

Toutanova, K., and Moore, R. 2002. Pronunciation modeling for improved spelling correction. In *ACL*, pp. 144-151.

Whitelaw, C., Hutchinson, B., Chung, G. Y., and Ellis, G. 2009. Using the web for language independent spellchecking and autocorrection. In *EMNLP*, pp. 890-899.

Zhang, Y., Hildebrand, Al. S., and Vogel, S. 2006. Distributed language modeling for n-best list reranking. In *EMNLP*, pp. 216-233.

RTG based surface realisation for TAG

Claire Gardent
CNRS/LORIA
claire.gardent@loria.fr

Laura Perez-Beltrachini
Université Henri Poincaré/LORIA
laura.perez@loria.fr

Abstract

Surface realisation with grammars integrating flat semantics is known to be NP complete. In this paper, we present a new algorithm for surface realisation based on Feature Based Tree Adjoining Grammar (FTAG) which draws on the observation that an FTAG can be translated into a Regular Tree Grammar describing its derivation trees. We carry out an extensive testing of several variants of this algorithm using an automatically produced testsuite and compare the results obtained with those obtained using GenI, another FTAG based surface realiser.

1 Introduction

As shown in (Brew, 1992; Koller and Striegnitz, 2002), Surface Realisation is NP-complete. Various optimisation techniques have therefore been proposed to help improve practical runtimes. For instance, (Kay, 1996) proposes to reduce the number of constituents built during realisation by only considering for combination constituents with non overlapping semantics and compatible indices. (Kay, 1996; Carroll and Oepen, 2005; Gardent and Kow, 2007) propose various techniques to restrict the combinatorics induced by intersective modifiers all applying to the same structure. And (Koller and Striegnitz, 2002; Gardent and Kow, 2007) describe two alternative techniques for reducing the initial search space.

In this paper, we focus on the optimisation mechanisms of two TAG based surface realisers namely, GENI (Gardent and Kow, 2007) and the algorithm we present in this paper namely, RTGEN (Perez-Beltrachini, 2009). GENI's optimisation includes both a filtering process whose aim is to reduce the initial search space and a two step, "substitution before adjunction", tree combination phase whose effect is to delay modifier adjunction thereby reducing the number of intermediate structures being built. In RTGEN on the other hand, the initial FTAG is converted to a Regular Tree Grammar (RTG) describing its derivation trees and an Earley algorithm, including sharing and packing, is used to optimise tree combination.

We compare GENI with several variants of the proposed RTGEN algorithm using an automatically produced testsuite of 2 679 input formulae and relate the RTGEN approach to existing work on surface realisation optimisation.

The paper is structured as follows. We first present the grammar used by both GENI and RTGEN, namely SEMXTAG (Section 2). We then describe the two surface realisation algorithms (Section 3). In Section 4, we describe the empirical evaluation carried out and present the results. Finally, Section 5 situates RTGEN with respect to related work on surface realisation optimisation.

2 SemXTag

The grammar (SEMXTAG) used by GENI and RTGEN is a Feature-Based Lexicalised Tree Adjoining Grammar (FTAG) augmented with a unification-based semantics as described in (Gardent and Kallmeyer, 2003). We briefly introduce each of these components and describe the grammar coverage. We then show how this FTAG can be converted to an RTG describing its derivation trees.

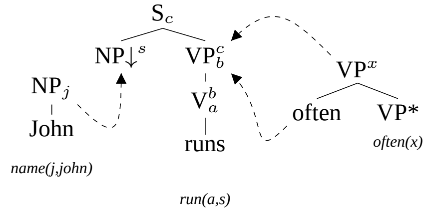
Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 367–375,
Beijing, August 2010

2.1 FTAG.

A Feature-based TAG (Vijay-Shanker and Joshi, 1988) consists of a set of (auxiliary or initial) elementary trees and of two tree-composition operations: substitution and adjunction. Initial trees are trees whose leaves are labeled with substitution nodes (marked with a downarrow) or terminal categories. Auxiliary trees are distinguished by a foot node (marked with a star) whose category must be the same as that of the root node. Substitution inserts a tree onto a substitution node of some other tree while adjunction inserts an auxiliary tree into a tree. In an FTAG, the tree nodes are furthermore decorated with two feature structures (called **top** and **bottom**) which are unified during derivation as follows. On substitution, the top of the substitution node is unified with the top of the root node of the tree being substituted in. On adjunction, the top of the root of the auxiliary tree is unified with the top of the node where adjunction takes place; and the bottom features of the foot node are unified with the bottom features of this node. At the end of a derivation, the top and bottom of all nodes in the derived tree are unified. Finally, each sentence derivation in an FTAG is associated with both a **derived tree** representing the phrase structure of the sentence and a **derivation tree** recording how the corresponding elementary trees were combined to form the derived tree. Nodes in a derivation tree are labelled with the name of a TAG elementary tree. Edges are labelled with a description of the operation used to combine the TAG trees whose names label the edge vertices.

2.2 FTAG with semantics.

To associate semantic representations with natural language expressions, the FTAG is modified as proposed in (Gardent and Kallmeyer, 2003).



$\Rightarrow$ *name(j,john), run(a,j), often(a)*

Figure 1: Flat Semantics for "John often runs"

Each elementary tree is associated with a flat semantic representation. For instance, in Figure 1,[1] the trees for *John*, *runs* and *often* are associated with the semantics *name(j,john)*, *run(a,s)* and *often(x)* respectively. Importantly, the arguments of a semantic functor are represented by unification variables which occur both in the semantic representation of this functor and on some nodes of the associated syntactic tree. For instance in Figure 1, the semantic index s occurring in the semantic representation of *runs* also occurs on the subject substitution node of the associated elementary tree. The value of semantic arguments is determined by the unifications resulting from adjunction and substitution. For instance, the semantic index s in the tree for *runs* is unified during substitution with the semantic index labelling the root node of the tree for *John*. As a result, the semantics of *John often runs* is {`name(j,john),run(a,j),often(a)`}.

2.3 SemXTAG.

SEMXTAG is an FTAG for English augmented with a unification based compositional semantics of the type described above. Its syntactic coverage approaches that of XTAG, the FTAG developed for English by the XTAG group (The XTAG Research Group, 2001). Like this grammar, it contains around 1300 elementary trees and covers auxiliaries, copula, raising and small clause constructions, topicalization, relative clauses, infinitives, gerunds, passives, adjuncts, ditransitives and datives, ergatives, it-clefts, wh-clefts, PRO constructions, noun-noun modification, extraposition, sentential adjuncts, imperatives and resultatives.

2.4 Converting SemXTAG to RTG

As shown in (Schmitz and Le Roux, 2008), an FTAG can be converted to a Regular Tree Grammar describing its derivation tree. In this section, we briefly sketch this conversion process. For a more precise description of this FTAG to RTG conversion, the reader is referred to (Schmitz and Le Roux, 2008).

[1] C^x/C_x abbreviate a node with category C and a top/bottom feature structure including the feature-value pair { **index :** x}.

In the FTAG-to-RTG conversion, each SEMXTAG elementary tree is converted to a rule that models its contribution to a TAG derivation tree. A TAG derivation involves the selection of an initial tree, which has some nodes requiring substitution and some permitting adjunction. Let us think of the potential adjunction sites as requiring, rather than permitting, adjunction, but such that the requirement can be satisfied by 'null' adjunction. Inserting another tree into this initial tree satisfies one of the substitution or adjunction requirements, but introduces some new requirements into the resulting tree, in the form of its own substitution nodes and adjunction sites.

Thus, intuitively, the RTG representation of a SEMXTAG elementary tree is a rule that rewrites the satisfied requirement as a local tree whose root is a unique identifier of the tree and whose leaves are the introduced requirements. A requirement of a substitution or adjunction of a tree of root category X is written as X_S or X_A, respectively. Here, for example, is the translation to RTG of the FTAG tree (minus semantics) for *run* in Figure 1, using the word anchoring the tree as its identifier (the upperscripts abbreviates features structures: b/t refers to the bottom/top feature structure and the upper case letters to the semantic index value, $[idx : X]$ is abbreviated to X):

$$S_S^{[t:T]} \rightarrow runs(S_A^{[t:T,b:C]}\ NP_S^{[t:S]}\ VP_A^{[t:C,b:B]}\ V_A^{[t:B,b:A]})$$

The semantics of the SemXTAG tree are carried over as-is to the corresponding RTG rule. Further, the feature structures labelling the nodes of the SemXTAG tree are converted into the RTG rules so as to correctly interact with substitution and adjunction (see (Schmitz and Le Roux, 2008) for more details on this part of the conversion process).

To account for the optionality of adjunction, there are additional rules allowing any adjunction requirement to be rewritten as the symbol ϵ, a terminal symbol of the RTG.

The terminal symbols of the RTG are thus the tree identifiers and the symbol ϵ, and its non-terminals are X_S and X_A for each terminal or non-terminal X of SemXTAG.

3 TAG-based surface realisation

We now present RTGEN and describe GENI, and compare the optimisations they propose to deal with the task complexity.

GENI and RTGEN are similar on several points. They use the same grammar, namely SEMXTAG (cf. Section 2). Further, they both pipeline three main steps. First, **lexical selection** selects from the grammar those elementary trees whose semantics subsumes part of the input semantics. Second, the **tree combining** phase systematically tries to combine trees using substitution and adjunction. Third, the **retrieval phase** extracts the yields of the complete derived trees, thereby producing the generated sentence(s).

GENI and RTGEN differ however with respect to the trees they are working with (derived trees in GENI *vs* derivation trees in RTGEN). They also differ in how tree combination is handled. We now describe these differences in more detail and explain how each approach address the complexity issue.

3.1 GenI

The tree combining phase in GENI falls into two main steps namely, filtering and tree combining.

Filtering. The so-called polarity filtering step aims to reduce the initial search space. It eliminates from the initial search space all those sets of TAG elementary trees which cover the input semantics but cannot possibly lead to a valid derived tree. In specific, this filtering removes all tree sets covering the input semantics such that either the category of a substitution node cannot be canceled out by that of the root node of a different tree; or a root node fails to have a matching substitution site. Importantly, this filtering relies solely on categorial information – feature information is not used. Furthermore, auxiliary trees have no impact on filtering since they provide and require the same category thereby being "polarity neutral elements".

Tree combining. The tree combining algorithm used after filtering has taken place, is a bottom-up tabular algorithm (Kay, 1996) optimised for TAGs. This step, unlike the first, uses all the features present in the grammar. To handle intersective modifiers, the delayed modifiers insertion strategy from (Carroll et al., 1999) is adapted to TAG as follows. First, all possible derived trees are obtained using only substitution. Next, adjunction is applied. Although the number of intermediate structures generated is still 2^n for n modifiers, this strategy has the effect of blocking these 2^n structures from multiplying out with other structures in the chart.

3.2 RTGen

RTGen synthesises different techniques that have been observed in the past to improve surface realisation runtimes. We first describe these techniques i.e., the main features of RTGEN. We then present three alternative ways of implementing RTGEN which will be compared in the evaluation.

3.2.1 RTGen's main features

A main feature of RTGEN is that it focuses on building derivation rather than derived trees. More specifically, the first two steps of the surface realisation process (lexical selection, tree combining) manipulate RTG rules describing the contribution of the SEMXTAG elementary trees to the derivation tree rather than the elementary tree themselves. The derived trees needed to produce actual sentences are only produced in the last phase i.e., the retrieval phase.

This strategy is inspired from a similar approach described in (Koller and Striegnitz, 2002) which was shown to be competitive with state of the art realisers on a small sample of example input chosen for their inherent complexity. (Koller and Striegnitz, 2002)'s approach combines trees using a constraint based dependency parser rather than an Earley algorithm so that it is difficult to assess how much of the efficiency is due to the parser and how much to the grammar conversion. Intuitively however, the motivation underlying the construction of a derivation rather than a derived tree is that efficiency might be increased because the context free derivation trees (i) are simpler than the mildly context sensitive trees generated by an FTAG and (ii) permit drawing on efficient parsing and surface realisation algorithms designed for such grammars.

Second, RTGEN makes use of the now standard semantic criteria proposed in (Kay, 1996; Carroll et al., 1999) to reduce the number of combinations tried out by the realiser. On the one hand, two constituents are combined by the algorithm's inference rules only if they cover disjoint parts of the input semantics. On the other hand, the semantic indices present in both the input formula and the lexically retrieved RTG trees are used to prevent the generation of intermediate structures that are not compatible with the input semantics. For instance, given the input formula for "John likes Mary", semantic indices will block the generation of "likes John" because this constituent requires that the constituent for "John" fills the patient slot of "likes" whereas the input semantics requires that it fills the agent slot. In addition, chart items in RTGEN are indexed by semantic indices to efficiently select chart items for combination.

Third, RTGEN implements a standard Earley algorithm complete with sharing and packing. Sharing allows for intermediate structures that are common to several derivations to be represented only once – in addition to not being recomputed each time. Packing means that partial derivation trees with identical semantic coverage and similar combinatorics (same number and type of substitution and adjunction requirements) are grouped together and that only one representative of such groups is stored in the chart. In this way, intermediate structures covering the same set of intersective modifiers in a different order are only represented once and the negative impact of intersective modifiers is lessened (cf. (Brew, 1992)). . As (Carroll and Oepen, 2005) have shown, packing and sharing are important factors in improving efficiency. In particular, they show that an algorithm with packing and sharing clearly outtperforms the same algorithm without packing and sharing giving an up to 50 times speed-up for inputs with large numbers of realizations.

3.2.2 Three ways to implement RTGen

Depending on how much linguistic information (i.e. feature constraints from the feature structures) is preserved in the RTG rules, several RTGEN configurations can be tried out which each reflect a different division of labour between constraint solving and structure building. To experiment with these several configurations, we exploit the fact that the FTAG-to-RTG conversion procedure developed by Sylvain Schmitz permits specifying which features should be preserved by the conversion.

RTGen-all. In this configuration, all the feature structure information present in the SEMXTAG elementary trees is carried over to the RTG rules. As a result, tree combining and constraint solving proceed simultaneously and the generated parse forest contains the derivation trees of all the output sentences.

RTGen-level0. In the RTGen-level0 configuration, only the syntactic category and the semantic features are preserved by the conversion. As a result, the grammar information used by the (derivation) tree building phase is comparable to that used by GENI filtering step. In both cases, the aim is to detect those sets of elementary trees which cover the input semantics and such that all syntactic requirements are satisfied while no syntactic resource is left out. A further step is additionally needed to produce only those trees which can be built from these tree sets when applying the constraints imposed by other features. In GENI, this additional step is carried out by the tree combining phase, in RTGEN, it is realised by the extraction phase i.e., the phase that constructs the derived trees from the derivation trees produced by the tree combining phase.

RTGen-selective. Contrary to parsing, surface realisation only accesses the morphological lexicon last i.e., after sentence trees are built. Because throughout the tree combining phase, lemmas are handled rather than forms, much of the morpho-syntactic feature information which is necessary to block the construction of ill-formed constituents is simply not available. It is therefore meaningful to only include in the tree combining phase those features whose value is available at tree combining time. In a third experiment, we automatically identified those features from the observed feature structure unification failures during runs of the realisation algorithm. We then use only these features (in combination with the semantic features and with categorial information) during tree combining.

4 Evaluation

To evaluate the impact of the different optimisation techniques discussed in the previous section, we use two benchmarks generated automatically from SEMXTAG (Gottesman, 2009).

The first benchmark (MODIFIERS) was designed to test the realisers on cases involving intersective modifiers. It includes 1 789 input formulae with a varying number (from 0 to 4 modifications), type (N and VP modifications) and distribution of intersective modifiers (n modifiers distributed differently over the predicate argument structures). For instance, the formula in (1) involves 2 N and 1 VP modification. Further, it combines lexical ambiguity with modification complexities, i.e. for the *snore* modifier the grammar provides 10 trees.

(1) $l_1 : \exists(x_1, h_r, h_s), h_r \geq l_2, h_s \geq l_3, l_2 : man(x_1), l_2 : snoring(e_1, x_1), l_2 : big(x_1), l_3 : sleep(e_2, x_1), l_4 : soundly(e_2)$
(*A snoring big man sleeps soundly*)

The second benchmark (COMPLEXITY) was designed to test overall performance on cases of differing complexity (input formulae of increasing length, involving verbs with a various number and types of arguments and with a varying number of and types of modifiers). It contains 890 distinct cases. A sample formula extracted from this benchmark is shown in (2), which includes one modification and to different verb types.

(2) $h_1 \geq l_4, l_0 : want(e, h_1), l_1 : \exists(x_1, h_r, h_s), h_r \geq l_1, h_s \geq l_0, l_1 : man(x_1), l_1 : snoring(e_1, x_1), l_3 : \exists(x_2, h_p, h_w, h_u), h_p \geq l_3, h_w \geq l_4, h_u \geq l_5, l_3 : monkey(x_2), l_4 : eat(e_2, x_2, e_3), l_5 : sleep(e_3, x_2)$
(*The snoring man wants the monkey to sleep*)

To evaluate GENI and the various configurations of RTGEN (RTGEN-all, RTGEN-level0, RTGEN-selective), we ran the 4 algorithms in batch mode on the two benchmarks and collected the following data for each test case:

- Packed chart size : the number of chart items built. This feature is only aplicable to RTGen as GENI does not implement packing.

- Unpacked chart size : the number of intermediate and final structures available after unpacking (or at the end of the tree combining process in the case of GENI).
- Initial Search Space (ISS) : the number of all possible combinations of elementary trees to be explored given the result of lexical selection on the input semantics. That is, the product of the number of FTAG elementary trees selected by each literal in the input semantics.
- Generation forest (GF) : the number of derivation trees covering the input semantics.

The graph in Figure 2 shows the differences between the different strategies with respect to the unpacked chart size metric.

A first observation is that RTGEN-all outperforms GENI in terms of intermediate structures built . In other words, the Earley sharing and packing strategy is more effective in reducing the number of constituents built than the filtering and substitution-before-adjunction optimisations used by GENI. In fact, even when no feature information is used at all (RTGEN-level0 plot), for more complex test cases, packing and sharing is more effective in reducing the chart size than filtering and operation ordering.

Another interesting observation is that RTGEN-all and RTGEN-selective have the same impact on chart size (their plots coincide). This is unsurprising since the features used by RTGEN-selective have been selected based on their ability to block constituent combination. The features used in RTGEN-selective mode are `wh`, `xp`, `assign-comp`, `mode`, `definite`, `inv`, `assign-case`, `rel-clause`, `extracted` and `phon`, in addition to the categorial and semantic information. In other words, using all 42 SEMXTAG grammar features has the same impact on search space pruning as using only a small subset of them. As explained in the previous section, this is probably due to the fact that contrary to parsing, surface realisation only accesses the morphological lexicon after tree combining takes place. Another possibility is that the grammar is under constrained and that feature values are missing thereby inducing overgeneration.

Zooming in on cases involving three modifiers, we show in Table 1 the average results for various efficiency metrics [2]. This provides a more detail view of the performance of the differences among the three RTGEN variants.

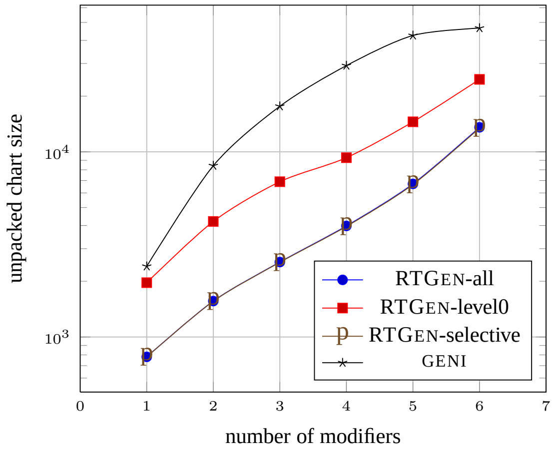


Figure 2: Performance of realisation approaches on the MODIFIERS benchmark, average unpacked chart size as a function of the number of modifiers.

strategy	GF	chart	unpacked-chart	seconds
RTGen-all	15.05	918.31	2,538.98	0.99
RTGen-level0	1,118.06	2,018	6,898.28	1.41
RTGen-selective	27.08	910.34	2,531.23	0.44

Table 1: Average results on 610 test cases from the MODIFIERS benchmark. Each test case has 3 modifications, distributed in various ways between adjectival and adverbial modifications. The second column, Generation Forest (GF), is the number of derivation trees present in the generated parse forest. The third and fourth columns show the chart and unpacked chart sizes, respectively. The last column shows the runtime in seconds.

This data shows that running RTGEN with no feature information leads not only to an increased chart size but also to runtimes that are higher in average than for full surface realisation i.e., realisation using the full grammar complete with con-

[2]The two realisers being implemented in different programming languages (RTGEN uses Prolog and GENI Haskell), runtimes comparisons are not necessarily very meaningful. Additionally, GENI does not provide time statistics. After adding this functionality to GENI, we found that overall GENI is faster on simple cases but slower on more complex ones. We are currently working on optimising RTGEN prolog implementation before carrying out a full scale runtime comparison.

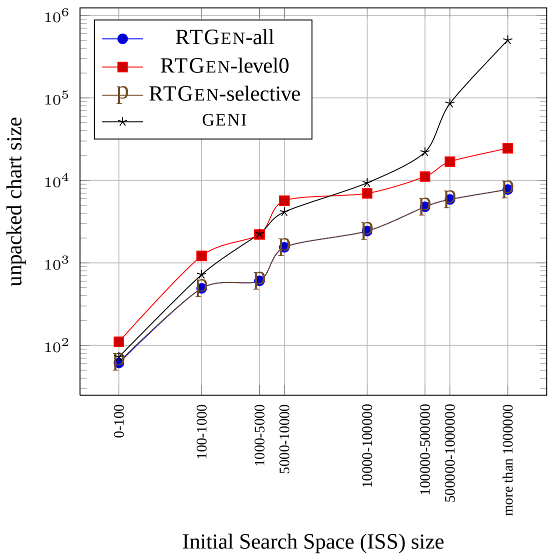


Figure 3: Performance of realisation approaches on the COMPLEXITY benchmark, average unpacked chart size as a function of the ISS complexity.

straints.

Interestingly, it also shows that the selective mode (RTGEN-selective) permits improving runtimes while achieving almost perfect disambiguation in that the average number of derivation trees (GF) produced is close to that produced when using all features. The differences between the two generation forests stems from packing. Using only a subset of features favors packing, thereby reducing the number of chart items built, but introduces over- generation.

Graph 3 and Table 2 confirm the results obtained using the MODIFIERS benchmark on a testset (COMPLEXITY) where input complexity varies not only with respect to modification but also with respect to the length of the input and to the degree of lexical ambiguity. Typically, in a TAG, one word or one semantic literal may be associated either with one tree or with up to several hundred trees (e.g., ditransitive verbs and verbs with several subcategorisation types). By varying the type and the number of verbs selected by the semantic literals contained in the input semantics, the COMPLEXITY benchmark provides a more extensive way to test performance on cases of varying complexity.

strategy	GF	chart	unpacked-chart	seconds
RTGen-all	14.77	693.39	2,427.82	0.81
RTGen-level0	162.02	2,114.16	6,954.84	1.09
RTGen-selective	15.31	692.9	2,427.2	0.36

Table 2: Average results on 335 cases with $10000 < ISS \leq 100000$, from the COMPLEXITY benchmark. The columns show the same performance metrics as in Table 1.

5 Related work

Much work has already been done on optimising surface realisation. Because surface realisation often draws on parsing techniques, work on parsing optimisation is also relevant. In this section, we briefly relate our proposal to another grammar converting approach (Koller and Striegnitz, 2002); to another chart based approach (Carroll and Oepen, 2005); and to approaches based on statistical pruning (White, 2004; Bangalore and Rambow, 2000).

5.1 Optimising surface realisation

Encoding into another grammatical formalism. As already mentioned, the RTGEN approach is closely related to the work of (Koller and Striegnitz, 2002) where the XTAG grammar is converted to a dependency grammar capturing its derivation trees. This conversion enables the use of a constraint based dependency parser, a parser which was specifically developed for the efficient parsing of free word order languages and is shown to support an efficient handling of both lexical and modifier attachment ambiguity.

Our proposal differs from this approach in three main ways. First, contrary to XTAG, SEMXTAG integrates a full-fledged, unification based compositional semantics thereby allowing for a principled coupling between semantic representations and natural language expressions. Second, the grammar conversion and the feature-based RTGs used by RTGEN accurately translates the full range of unification mechanisms employed in FTAG wheras the conversion described by (Koller and Striegnitz, 2002) does not take into account feature structure information. Third, the RTGEN approach was extensively tested on a large benchmark using 3 different configurations whilst (Koller and Striegnitz, 2002) results are restricted to a few hand constructed example inputs.

Chart generation algorithm optimisations. (Carroll and Oepen, 2005) provides an extensive and detailed study of how various techniques used to optimise parsing and surface realisation impact the efficiency of a surface realiser based on a large coverage Head-Driven Phrase Structure grammar.

Because they use different grammars, grammar formalisms and different benchmarks, it is difficult to compare the RTGEN and the HPSG approach. However, one point is put forward by (Carroll and Oepen, 2005) which it would be interesting to integrate in RTGEN(Carroll and Oepen, 2005) show that for packing to be efficient, it is important that equivalence be checked through subsumption, not through equality. RTGEN also implements a packing mechanism with subsumption check, i.e. different ways of covering the same subset of the input semantics are grouped together and represented in the chart by the most general one. One difference however it that RTGEN will pack analyses together as long as the new ones are more specific cases. It will not go backwards to recalculate the packing made so far if a more general item is found (Stefan and John, 2000). In this case the algorithm will pack them under two different groups.

Statistical pruning. Various probabilistic techniques have been proposed in surface realisation to improve e.g., lexical selection, the handling of intersective modifiers or ranking. For instance, (Bangalore and Rambow, 2000) uses a tree model to produce a single most probable lexical selection while in White's system, the best paraphrase is determined on the basis of n-gram scores. Further, to address the fact that there are $n!$ ways to combine any n modifiers with a single constituent, (White, 2004) proposes to use a language model to prune the chart of identical edges representing different modifier permutations, e.g., to choose between *fierce black cat* and *black fierce cat*. Similarly, (Bangalore and Rambow, 2000) assumes a single derivation tree that encodes a word lattice (*a* {*fierce black, black fierce*} *cat*), and uses statistical knowledge to select the best linearisation. Our approach differs from these approaches in that lexical selection is not filtered, intersective modifiers are handled by the grammar (constraints on the respective order of adjectives) and the chart packing strategy (for optimisation), and ranking is not performed. We are currently exploring the use of Optimality Theory for ranking.

6 Conclusion

We presented RTGEN, a novel surface realiser for FTAG grammars which builds on the observation that an FTAG can be translated to a regular tree grammar describing its derivation trees. Using automatically constructed benchmarks, we compared the performance of this realiser with that of GENI, another state of the art realiser for FTAG. We showed that RTGEN outperforms GENI in terms of space i.e. that the Earley sharing and packing strategy is more effective in reducing the number of constituents built than the filtering and substitution-before-adjunction optimisations used by GENI. Moreover, we investigated three ways of interleaving phrase structure and feature structure constraints and showed that, given a naive constraint solving approach, the interleaving approach with selective features seems to provide the best space/runtimes compromise.

Future work will concentrate on further investigating the interplay in surface realisation between phrase structure and feature structure constraints. In particular, (Maxwell and Kaplan, 1994) shows that a more sophisticated approach to constraint solving and to its interaction with chart processing renders the non interleaved approach more effective than the interleaved one. We plan to examine whether this observation applies to SEMXTAG and RTGEN. Further, we intend to integrate Optimality Theory constraints in RTGEN so as support ranking of multiple outputs. Finally, we want to further optimise RTGEN on intersective modifiers using one the methods mentioned in Section 5.

References

Bangalore, S. and O. Rambow. 2000. Using TAGs, a tree model and a language model for generation. In *Proceedings of TAG+5*, Paris, France.

Brew, Chris. 1992. Letting the cat out of the bag: generation for shake-and-bake mt. In *Proceedings* *of the 14th conference on Computational linguistics*, pages 610–616, Morristown, NJ, USA. Association for Computational Linguistics.

Carroll, J. and S. Oepen. 2005. High efficiency realization for a wide-coverage unification grammar. *2nd IJCNLP*.

Carroll, J., A. Copestake, D. Flickinger, and V. Paznański. 1999. An efficient chart generator for (semi-)lexicalist grammars. In *Proceedings of EWNLG '99*.

Gardent, C. and L. Kallmeyer. 2003. Semantic construction in FTAG. In *10th EACL*, Budapest, Hungary.

Gardent, C. and E. Kow. 2007. Spotting overgeneration suspect. In *11th European Workshop on Natural Language Generation (ENLG)*.

Gottesman, B. 2009. Generating examples. Master's thesis, Erasmus Mundus Master Language and Communication Technology, Saarbrucken/Nancy.

Kay, Martin. 1996. Chart generation. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 200–204, Morristown, NJ, USA. Association for Computational Linguistics.

Koller, A. and K. Striegnitz. 2002. Generation as dependency parsing. In *Proceedings of the 40th ACL*, Philadelphia.

Maxwell, J. and R. Kaplan. 1994. The interface between phrasal and functional constraints. *Computational Linguistics*, 19(4).

Perez-Beltrachini, L. 2009. Using regular tree grammars to reduce the search space in surface realisation. Master's thesis, Erasmus Mundus Master Language and Communication Technology, Nancy/Bolzano.

Schmitz, S. and J. Le Roux. 2008. Feature unification in TAG derivation trees. In Gardent, C. and A. Sarkar, editors, *Proceedings of the 9th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+'08)*, pages 141–148, Tübingen, Germany.

Stefan, Oepen and Carroll John. 2000. Parser engineering and performance profiling. *Journal of Natural Language Engineering*, 6(1):81–98.

The XTAG Research Group. 2001. A lexicalised tree adjoining grammar for english. Technical report, Institute for Research in Cognitive Science, University of Pennsylvannia.

Vijay-Shanker, K. and AK Joshi. 1988. Feature Structures Based Tree Adjoining Grammars. *Proceedings of the 12th conference on Computational linguistics*, 55:v2.

White, M. 2004. Reining in CCG chart realization. In *INLG*, pages 182–191.

Automatically Learning Source-side Reordering Rules for Large Scale Machine Translation

Dmitriy Genzel
Google, Inc.
dmitriy@google.com

Abstract

We describe an approach to automatically learn reordering rules to be applied as a preprocessing step in phrase-based machine translation. We learn rules for 8 different language pairs, showing BLEU improvements for all of them, and demonstrate that many important order transformations (SVO to SOV or VSO, head-modifier, verb movement) can be captured by this approach.

1 Introduction

One of the major problems of modern statistical machine translation relates to its difficulties in producing the correct word order on the target side of the translation where the source side order is not the same as the target side. In many cases where the translation is spectacularly bad, if one only enters the source sentence in the word order of the target language the translation becomes near-perfect (largely because the language model can now make sense of it). The word order problems are especially extensive for languages that have major differences, such as SOV vs. SVO languages, but also cause insidious, but entirely avoidable errors for the language pairs where the word order is almost right, but not quite[1]. For practical reasons all phrase-based decoders limit the amount of reordering allowed and thus are completely unable to produce correct translations when the necessary movement is over a large distance. Furthermore, where the actual systematic reordering for the two languages is within the decoder's search space, it is penalized just as any other kind of reordering, whereas doing anything other than this systematic reordering should in fact be penalized.

It has been argued that this is a fundamental flaw in phrase-based decoding systems and hierarchical and syntax-based systems have been proposed to solve this problem. These systems can in principle resolve a part of this problem, but at a significant time cost during training, and even worse, during translation, making it less practical for realtime systems. Instead we propose a system for learning pre-ordering rules automatically from data and demonstrate that it can capture many different kinds of reordering phenomena and do so at no additional online cost.

[1]For example of the latter kind, verb movement for English-German and similar language pairs often causes verbs to be aligned to nothing and to be altogether dropped in translation.

2 Related Work

Many solutions to the reordering problem have been proposed, e.g. syntax-based models (Chiang, 2005), lexicalized reordering (Och et al., 2004), and tree-to-string methods (Zhang et al., 2006). All these methods try to solve the reordering problem in different ways, but have the following problems in common: word alignment is not affected by them and they tend to introduce significant additional work to be done at translation time. Most state of the art systems use HMM or IBM Model 4 word alignment, both of which have a penalty term associated with long distance jumps, and tend to misalign words which move far from their expected positions.

We are going to focus on the approaches where reordering is done as a preprocessing step (sometimes called pre-ordering). These approaches have the advantage that they are independent of the actual MT system used, are often fast to apply, and tend to decrease (due to improved quality of heuristic estimates) rather than dramatically increase the time spent in actual decoding, unlike

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 376–384, Beijing, August 2010

some of the previously mentioned approaches. The downside of these methods is that the reordering is fixed, and if it is wrong it can hurt the quality of translations. We will discuss solutions for this problem later.

Even in the relatively limited space of preprocessing-based reordering solutions, there has been a large amount of previous work, as far back as Brown et al. (1992). Most approaches focus on utilizing manually written rules for different languages. A common language pair for which rules were proposed is German-English (Nießen and Ney, 2001; Collins et al., 2005). There is similar work for Chinese-English (Wang et al., 2007) and quite a few other languages. Clearly, such methods work quite well, but require linguistic expertise to produce. Our goal, however, is to learn reordering from parallel data that is already available to an MT system in an entirely unsupervised manner.

We are not the first to attempt this task. In particular, Xia and McCord (2004) proposed a way to automatically learn reordering patterns for French-English. Their system parses parallel data both on the source and target side and then uses a variety of heuristics to extract reordering rules which are then applied during training. More recently, Li et al. (2007) use a maximum entropy system to learn reordering rules for binary trees (i.e., whether to keep or reorder for each node). An approach most similar to ours is that of Rottmann and Vogel (2007) where they learn reordering rules based on sequences of part-of-speech tags (but do not use parse trees). All of these approaches show improvements in translation quality, but are applied on a single language pair. Our goal is to find a method that works well for many language pairs, regardless of the word order transformations needed, and without language-specific tuning. Unlike our predecessors, we use a systematic search through the space of possible permutation rules to minimize a specific metric, related to the monotonicity of resulting alignments.

3 Our Approach

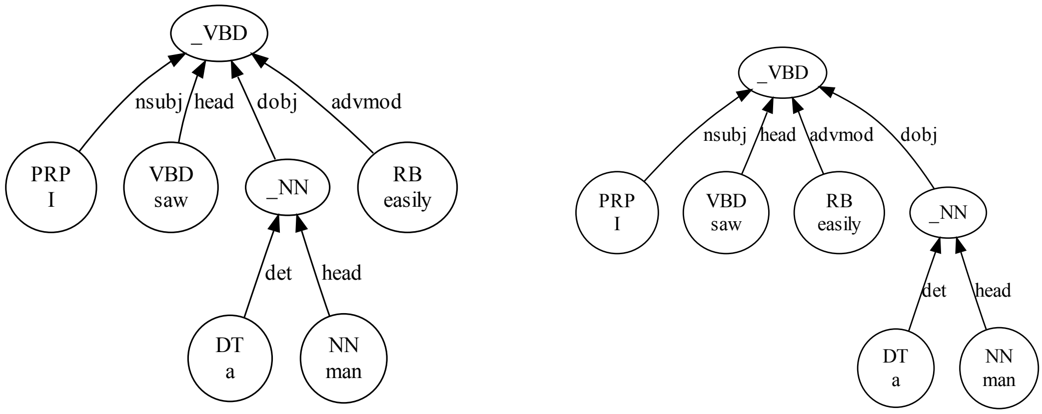
We limit ourselves to reorderings of the source side of training and test data. To constrain our reorderings, we first produce a parse tree, using a dependency parser similar to that of Nivre and Scholz (2004). The above parser is much faster than the time spent in translating the same sentence and thus creates almost no overhead. In our experiments where the source language is English the training data for the parser is the Penn Treebank (Marcus et al., 1993). For German, we use TIGER treebank (Brants et al., 2002). We then convert the dependency tree to a shallow constituent tree. The trees are annotated by both Penn Treebank part of speech tags and by Stanford dependency types (de Marneffe et al., 2006; de Marneffe and Manning, 2008). For an example, see Figure 1a.

Our reorderings are constrained by reordering of nodes in a parse tree of the source sentence. Thus, the full space of reorderings we consider consists of all reorderings that would produce a parse tree with the same set of child-parent relationships. For an example of a valid reordering, see Figure 1b.

Each reordering is described by a series of rules and we learn one such series for each language pair automatically. Each source sentence is parsed, and the tree is transformed sequentially, one rule at a time applying to the entire tree, top down. The reordered sentence is read off the leaves of the tree and training and evaluation proceeds as normal. We are using a state-of-the-art phrase-based statistical machine translation system to perform the actual translation. The system is itself capable of further local reordering during translation limited by the maximum distance of 4 words.

3.1 Rule Space

Each rule consists of two parts: conditioning context and action. For every internal node in the parse tree, traversed top-down, the node is matched against the conditioning context, and if a match is found, the associated action applies. All actions are limited to reordering children of the matching node. Furthermore, if a rule applies at a node, its descendants are not traversed for the purpose of matching to avoid modifying the same part of the sentence twice by the same rule. A different rule may apply on this node or its descendants



(a) A sample parse tree

(b) After reordering (moving RB over _NN)

Figure 1: Parse tree of a sentence and its reordering

Feature	Description
nT	POS tag of this node
nL	Syntactic label of this node
pT	POS tag of the parent of this node
pL	Syntactic label of the parent
1T	POS tag of the first child
1L	Label of the first child
2T	POS tag of the second child
2L	Label of the second child
...	...

Table 1: Set of features used as conditioning variables

later in the sequence.

A conditioning context is a conjunction of conditions. Each condition is a (feature, value) pair. List of features is given in table 1. In practice, we limit ourselves to no more than 4 conditions in a given context to avoid combinatorial explosion and sparsity as well as contexts that fail to generalize. However, we may exhaustively generate every possible conjunction of up to 5 conditions from this list that covers up to 4 children that we actually observe in training.

For example, the following contexts would be valid for transformation in Fig. 1:

- nT = _VBD
- 1T = PRP
- 1L = nsubj
- 3T = dobj
- etc.

or any conjunction of these. The action performed in this example is swapping children 3 and 4 of the _VBD node, and can be denoted as the permutation (1,2,4,3).

When processing a rule sequence, once a rule applies, the action is performed, and that rule is no longer applied on the same node or its descendants (but can be further applied elsewhere in the tree). Another rule (even an identical one) starts from the top and can apply to nodes modified by previous rules.

3.2 Reordering metrics

To evaluate the quality of a given reordering rule, we need to have reliable metrics that, for each sentence pair, can evaluate whether an improvement in monotonicity has been made.

The easiest metric to use is the number of crossing alignment links for a given aligned sentence pair. For instance, in Figure 2, there are 2 crossing links. This metric is trivial to compute and has some nice properties. For instance, moving a single word one position out of place causes one link

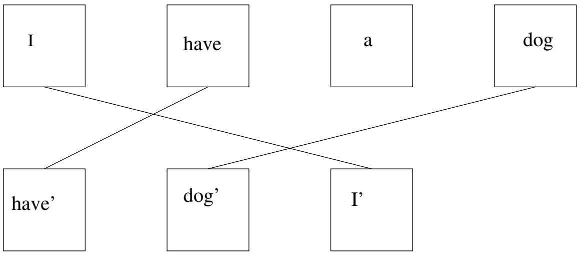


Figure 2: Counting crossing alignment links

to cross, moving it farther away from its correct position would cause more links to cross. We will refer to this metric as *crossing score*.

An ideal metric would be the actual BLEU score that the system would obtain under this reordering rule on the development set. However, since each rule affects word alignment, phrase extraction, optimal feature weights, and the actual translation, it would be necessary to retrain the entire phrase-based system for each possible rule, which is impractical. It is, however, practical, to retranslate the development set, keeping the phrase table and feature weights constant. Normally, however, phrase tables contain multi-word phrases, such as "a b" which may no longer match after the reordering, and this biases the system toward the original word order. To avoid this, for this computation only, we use a phrase table that only contains single words and is therefore independent of the source sentence word order. This lets us test whether a given reordering improves the search space for the phrase-based decoder at the relatively small computational cost of translating the development set. We obtain a difference of the BLEU scores with and without a given rule, which we hope to be a reasonable estimate of the true gain in BLEU score that one would obtain, by retraining the full system, including word alignment, full-length phrase extraction, and tuning the feature weights. We refer to this score as *estimated BLEU gain*.

Note that these two scores are used to obtain an estimate of utility of any given rule, and are not used for evaluation of the entire system. Those metrics are discussed in detail in the evaluation section.

3.3 Algorithm

We propose a straightforward algorithm to automatically learn reordering rules. The input data for all algorithms is word-aligned sentence pairs. We have found that sophisticated alignment models introduce a bias toward alignment between certain kinds of nodes (usually ones that are close), and this has undesirable effects. In practical terms this means that neither HMM nor Model 4 alignments are useful (even though they are better as alignments), but Model 1 alignments are. However, to compensate for poor quality of the alignments, we simply delete those alignment links that have posterior probabilities under 0.5[2] and remove sentence pairs which have very few alignments left. The crossing score works quite well even when only a portion of the words in a sentence are aligned.

The algorithm's outline is given as Alg. 1.

The algorithm proceeds by considering all rules after the best sequence of rules so far, and appends the best new rule (according to the metric) to the sequence. In practice, some changes are needed, and we describe some variations. Each of these variations produces a different sequence of rules, but they are interchangeable, and we can simply pick one that performs best on the development set, or to combine them through multi-source translation or consensus.

In all variations, we are unable to generate all possible rules for every sentence, as the number can easily be 10^4-10^6 per sentence. It is sufficient, however, to take a random sample of the input, extract top candidates, and reevaluate those on the entire set.

We also limit the kinds of rules we are allowed to generate. The number of possible actions on a node with n children is $n! - 1$ and our trees are quite shallow, often containing 5, 6, or even more children per node. To avoid dealing with explosion of rules and the resulting sparsity of the rule space, we modify the process slightly, so that instead of matching a node, we match a node and a consecutive subsequence of its children of a given size, as a sliding window. For example, in Figure 1a, node _VBD has 4 children. If we limit our-

[2]This guarantees only one alignment per word

Algorithm 1 Optimizing alignment links

```
input: A set of aligned sentence pairs
base = <empty sequence>;
for several iterations do
   candidate_rules = GenerateAllCandidateRules(input, base);
   base.append(MinCost(candidate_rules))
end for
```

selves to 3 children at a time we would attempt to match this node twice: with its children 1,2,3 and 2,3,4. In other words, we pretend to consider two nodes, one with the first set of children, and one with the second, proceeding left to right. If either one matches, we apply the action to the subset of children in the window and stop processing the node further.

It is also useful to produce more than one rule per iteration, although this can be problematic, since the rules may interfere with each other.

3.3.1 Variant 1: Optimizing crossing score

We start with the initially empty base sequence. As described above, we generate every possible rule from a subset of sentences, and evaluate them on the entire input, with the base sequence always applied first. We use crossing score as a metric. However, instead of extracting only one best-scoring rule, we extract K best. Now we need to obtain a decorrelated set: for every pair of rules, we count the number of sentences where they both apply. For every rule we consider all rules that are ranked higher, and if the percentage of matches between these two rules is high, the rules may interfere with each other, and the current rule is dropped. We thus obtain a small ordered set of rules that tend to apply on different sentences, and should not interfere with each other. From this ordered set we produce all candidate rule subsequences and evaluate them, to ensure there really is no interference. The one with the best score is then appended to the base sequence. The process is then repeated with a new base sequence.

3.3.2 Variant 2: Optimizing Estimated BLEU gain

We proceed as in the previous variant, but final evaluation of potential sequences to be appended is done differently. Instead of using a crossing score, we reorder the development set with each candidate rule sequence and score it using a translation system with a fixed phrase table with single word phrases only (to avoid bias for a specific word order). The sequence with the highest BLEU is then appended to base sequence, and the process is repeated.

3.3.3 Variant 3: Optimizing Estimated BLEU gain in sequence

In this variant, once we obtain a set of decorrelated candidate rules $\{a_1, a_2, \dots a_n\}$ ordered by crossing score, we evaluate the following rule sequences (where b is base sequence): $(b), (b, a_1), (b, a_1, a_2) \dots (b, a_1, \dots a_n)$ using estimated BLEU gain, as above. If we find that for some k, $score(b, a_1, \dots a_{k-1}) > score(b, a_1, \dots a_{k-1}, a_k)$, that means that a_k interferes with preceding rules. We remove all such a_k, and retranslate/rescore until the score sequence is monotonically non-decreasing. At this point, we append all surviving rules to the base sequence, and repeat the process.

4 Evaluation

As described above, our base system is a phrase-based statistical MT system, similar to that of Och and Ney (2004). The baseline decoder is capable of local reordering of up to 4 words. Our training data is extracted by mining from the Web, as well as from other published sources. We train systems from English to 7 other languages, as well as German-English. We chose them as follows: SOV languages (Japanese, Korean, Hindi), VSO language (Welsh), long distance verb movement (German), noun-modifier issues (Russian and Czech). The amount of training data varies from 28 million words (for Hindi) to 260 million (for German). The baseline system is a production-quality system used by a large number of users.

For the first set of experiments for German-English and English-German we use WMT-09 data sets for development and testing (Callison-Burch et al., 2009). We report BLEU scores for each of the algorithms along with the best score from the WMT-09 workshop for reference in Table 2.

Unfortunately, there is no standard data set for most of the languages we would like to experiment with. For the second set of experiments, we use an unpublished data set, containing data in English and 7 languages mentioned above. Our test data comes from two sources: news articles from WikiNews[3] (996 sentences) and a set of random sentences from the web (9000 sentences). From these, we create 3 sets: *dev1*: 3000 sentences from *web* and 486 sentences from *wiki*; *dev2*: 1000 sentences from *web*; and *test*: the remainder of *web* (5000 sentences) and *wiki* (510 sentences). The *dev1* set is used for tuning the system, both *dev1* and *dev2* for tuning consensus, and the *test* set for evaluation. These sets are the same for all 7 languages.

Discriminative minimum error rate training (Macherey et al., 2008) was applied to optimize the feature weights for each system.

We evaluate the three variants of the algorithm mentioned above. Each algorithm outputs a reordering rule sequence (40-50 rules long) which is applied to all the training and test data, and a complete system is trained from scratch.

There is no need for us to pick a single algorithm for all language pairs, since each algorithm produces rules that are compatible with each other. We are able to pick the algorithm that works best on the development set for each language pair.

In addition, we can use a decoder that is capable of performing a multi-input translation which is given the unreordered input as well as the three reordered inputs produced by the above algorithm. This decoder is able to learn separate feature weights for each feature/algorithm combination.

Finally, we can use consensus translation (Macherey and Och, 2007) to produce the best possible translation for each sentence.

[3]http://en.wikinews.org

Table 4: Manual vs. automatic reordering. *Automatic* score is the combined score from Table 3.

Language	Base	Manual	Auto-matic	Diff
Hindi	16.85	19.25	19.36	0.11
Japanese	25.91	28.78	29.12	0.34
Korean	23.61	27.99	27.91	-0.08

Results using BLEU score (character-level for Japanese and Korean, word-level for other languages) for English to X systems are given in Table 3, along with the score of Google Translate as of Feb 15, 2010, for expected quality reference. All gains in the combined and consensus columns are statistically significant using a bootstrap resampling test (Noreen, 1989).

We should also note that the parsing and reordering overhead was an average of 10msec per sentence, and had no appreciable impact on the speed of the system.

4.1 Comparison with manual reordering

We also compared our automatic method with a manually written reordering rule set for SOV languages (Xu et al., 2009) (rules initially written for Korean) for comparison with our approach. The results are given in Table 4. The results are mostly comparable, with automatic rules being better for two of the three languages.

4.2 Turning off decoder reordering

All of the above experiments allowed the decoder to further reorder the sentence as needed. Reordering in the decoder creates an exponential increase in the search space, and for a typical decoding strategy can lead to increase in decoding time, search errors, or both. Since we already preorder the sentence, it should be possible to avoid reordering in the decoder altogether.

Results for the combined decoder are given in Table 5. It contains the gain of the combined decoder against the baseline from Table 3, and the gain when decoder reordering is turned off against the same baseline (which has decoder reordering on). For many languages it is indeed now possi-

Table 2: Results for 3 algorithms on WMT-09 data with best individual system score from the workshop: for EN to DE, Edinburgh, for DE to EN, Google

Language	Base	Var. 1	Var. 2	Var. 3	Best workshop
EN to DE	16.09	16.30	16.35	**16.40**	*14.76*
DE to EN	21.00	**22.45**	22.13	22.05	*20.23*

Table 3: Results on internal test set for 3 systems (Variant 1,2,3), the variant which performed best on the development set, the combined system, and the consensus run, along with Google Translate scores (Feb 15, 2010) for reference

Language	Google	Base	Var. 1	Var. 2	Var. 3	Best on dev	Combined	Consensus
	%BLEU	%BLEU	gain	gain	gain	gain	gain	gain
Czech	16.68	15.35	-0.08	0.13	**0.19**	0.19	0.21	0.21
German	20.34	18.65	**0.47**	0.30	0.39	0.39	0.72	0.73
Hindi	19.15	16.85	**2.25**	2.08	0.15	2.08	2.51	2.47
Japanese	30.74	25.91	**3.05**	2.60	**3.05**	3.05	3.21	3.03
Korean	27.99	23.61	3.34	3.77	**4.16**	4.16	4.30	4.30
Russian	16.80	15.33	0.08	**0.10**	**0.10**	0.08	0.14	0.23
Welsh	27.38	25.48	1.25	0.77	**1.43**	1.43	1.34	1.63

Table 5: Disallowing decoder reordering: difference against baseline in %BLEU gain

Language	Decoder reordering	No decoder reordering
Czech	0.21	0.08
German	0.72	0.55
Hindi	2.51	2.27
Japanese	3.21	3.21
Korean	4.30	4.15
Russian	0.14	-0.10
Welsh	1.34	0.98

ble to avoid decoder reordering altogether which leads to a significant speedup.

5 Analysis

We looked at the rules being learned as well as at the differences in the output to see if the gains in BLEU are in fact due to the reordering phenomena being resolved. The top rules for each language are given in Table 6.

One can observe that the top rules for German and Slavic languages are as expected: verb movement and noun modifier reordering. Other top rules for German cover other specific cases of verb movement, other rules for Czech include, for example, movement of the subject of the passive sentence to the right and movement of the possessive (which is similar to the noun compound case).

The rules for Welsh include movement of the adjective modifier over its head (given in the table above) and other rules moving noun modifiers, moving a modal verb left over its subject, moving determiners to the right of nouns, etc.

For Japanese and Korean, there are many rules with dramatic impact, such as a rule moving all heads to the right, reversing a sequence of three nodes starting with a modal (e.g. *can do something* to *something do can*), moving numerical modifiers to the right of their heads, and many others.

Hindi is also an SOV language, but its grammar is not as similar to Japanese or Korean as they are to each other. Still, Hindi also has some similar rules, but there are many more involving verb movement, such as a rule directly moving the verb to the final position.

By looking at the sentences produced by the system we can see that the differences are dramatic for SOV and VSO languages, as expected,

Table 6: Examples of top rules and their application

Languages	Context	Order	Example
Hindi	1L:head 3L:none	2,1,3	*I see him → I him see*
Japanese, Korean	2L:prep	2,1	*eat with a spoon → eat a spoon with*
German	1T:VBN 2L:prep	2,1	*struck with a ball → with a ball struck*
Russian, Czech	1L:nn 2L:head	2,1	*a building entrance → a entrance building*
Welsh	1L:amod 2L:head	2,1	*blue ball → ball blue*

but more interestingly, most German sentences now have a verb where the baseline had none. Another profound effect can be observed for Russian: the baseline almost invariably translated noun compounds incorrectly: e.g. *group leaders* may be translated as *group of-leaders* since this requires no reordering and no preposition insertion. This is especially problematic, since the user of the translation system often cannot detect this: the resulting sentence is not ungrammatical and can even make sense. Our algorithm learns a rule that prevents this from happening. Now the decoder must pay a cost to keep the order the same as in English.

6 Discussion and Future Work

We have demonstrated a general technique which requires only access to a parser for the source language (in addition to parallel data which already exists for an MT system) and is capable of reducing reordering problems endemic in a phrase-based system. No linguists or even native speakers of any of these languages were needed to write the rules. The algorithm is quite robust and performs well on noisy web data, much of it being ungrammatical.

All variants turned out to perform well, although variants 1 and 3 were better most of the time. We consider all variants to be useful, since they find different local maxima under different objective functions, and in practice use all of them and pick a rule sequence that performs best on the development set for any specific language pair.

We plan to explore this research area further in several ways. First, it would be interesting to experiment with applying rules learned for one language to a related language, e.g. Portuguese for Spanish or German for Dutch. This would let us use rules learned from a major language for a minor one with less available training data.

We have only used English and German as source languages. There is training data for parsers in other languages, and this approach should work well for most source languages. Where a source language parser is not available, we can still improve quality, by learning rules from the target side and applying them only for the purpose of improving word alignment. Improving word alignment alone would not help as much as also using the reordering in the decoder, but it will probably help in extracting better phrases. We also plan to use parser projection to induce a reasonable quality parser for other languages.

References

Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The tiger treebank. In *In Proceedings of the Workshop on Treebanks and Linguistic Theories*, pages 24–41.

Peter F. Brown, Stephen A. Della, Pietra Vincent, J. Della Pietra, John D. Lafferty Robert, and L. Mercer. 1992. Analysis, statistical transfer, and synthesis in machine translation. In *Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 83–100.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece, March. Association for Computational Linguistics.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the ACL'05*, pages 263–270, Ann Arbor, Michigan, June.

Michael Collins, Philipp Koehn, and Ivona Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the ACL'05*, pages 531–540, Ann Arbor, Michigan, June.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. The Stanford typed dependencies representations. In *COLING'08 Workshop on Cross-framework and Cross-domain Parser Evaluation*, Manchester, England, August.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure trees. In *LREC*.

Chi-Ho Li, Minghui Li, Dongdong Zhang, Mu Li, Ming Zhou, and Yi Guan. 2007. A probabilistic approach to syntax-based reordering for statistical machine translation. In *Proceedings of the ACL-07*, pages 720–727, Prague, Czech Republic, June.

Wolfgang Macherey and Franz J. Och. 2007. An empirical study on computing consensus translations from multiple machine translation systems. In *Proceedings of the EMNLP-CoNLL'07*, pages 986–995, Prague, Czech Republic, June.

Wolfgang Macherey, Franz Och, Ignacio Thayer, and Jakob Uszkoreit. 2008. Lattice-based minimum error rate training for statistical machine translation. In *Proceedings of the EMNLP-2008*, pages 725–734, Honolulu, Hawaii, October.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Sonja Nießen and Hermann Ney. 2001. Morpho-syntactic analysis for reordering in statistical machine translation. In *Machine Translation Summit*, pages 247–252, Santiago de Compostela, Spain, September.

Joakim Nivre and Mario Scholz. 2004. Deterministic dependency parsing of English text. In *Proceedings of Coling 2004*, pages 64–70, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Eric W. Noreen. 1989. *Computer-Intensive Methods for Testing Hypotheses*. John Wiley & Sons, Canada.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. In *HLT-NAACL 2004: Main Proceedings*, pages 161–168, Boston, Massachusetts, USA, May 2 - May 7.

Kay Rottmann and Stephan Vogel. 2007. Word reordering in statistical machine translation with a pos-based distortion model. In *Proceedings of TMI*, Skovde, Sweden.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the EMNLP-CoNLL'2007*, pages 737–745, Prague, Czech Republic, June.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of Coling 2004*, pages 508–514, Geneva, Switzerland, Aug 23–Aug 27. COLING.

Peng Xu, Jaeho Kang, Michael Ringgaard, and Franz Och. 2009. Using a dependency parser to improve SMT for subject-object-verb languages. In *Proceedings of NAACL-HLT'09*, Boulder, Colorado.

Hao Zhang, Liang Huang, Daniel Gildea, and Kevin Knight. 2006. Synchronous binarization for machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 256–263, New York City, USA, June. Association for Computational Linguistics.

Comparing Language Similarity across Genetic and Typologically-Based Groupings

Ryan Georgi
University of Washington
rgeorgi@uw.edu

Fei Xia
University of Washington
fxia@uw.edu

William Lewis
Microsoft Research
wilewis@microsoft.com

Abstract

Recent studies have shown the potential benefits of leveraging resources for resource-rich languages to build tools for similar, but resource-poor languages. We examine what constitutes "similarity" by comparing traditional phylogenetic language groups, which are motivated largely by genetic relationships, with language groupings formed by clustering methods using typological features only. Using data from the World Atlas of Language Structures (WALS), our preliminary experiments show that typologically-based clusters look quite different from genetic groups, but perform as good or better when used to predict feature values of member languages.

1 Introduction

While there are more than six thousand languages in the world, only a small portion of these languages have received substantial attention in the field of NLP. With the increase in use of data-driven methods, languages with few or no electronic resources have been difficult to process with current methods. The morphological tagging of Russian using Czech resources as done by (Hana et al., 2004) shows the potential benefit for using the resources of resource-rich languages to bootstrap NLP tools for related languages. Projecting syntactic structures across languages (Yarowsky and Ngai, 2001; Xia and Lewis, 2007) is another possible way to harness existing tools, though such projection is more reliable among languages with similar syntax.

Studies such as these show the possible benefits of working with similar languages. A crucial question is how we should define similarity between languages. While genetically related languages tend to have similar typological features as they could inherit the features from their common ancestor, they could also differ a lot due to language change over time. On the other hand, languages with no common ancestor could share many features due to language contact and other factors.

It is worth noting that the goals of historical linguistics differ from those of language typology in that while historical linguistics focuses primarily on diachronic language change, typology is more focused on a synchronic survey of features found in the world's languages: what typological features exist, where they are found, and why a language has a feature.

These differences between the concepts of genetic relatedness and language similarities lead us to the following questions:

Q1. If we cluster languages based only on their typological features, how do the induced clusters compare to phylogenetic groupings?

Q2. How well do induced clusters and genetic families perform in predicting values for typological features?

Q3. What typological features tend to stay the same within language families, and what features are likely to differ?

These questions are the focus of this study, and for the experiments, we use information from World Atlas of Language Structures (Haspelmath et al., 2005), or WALS.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 385–393,
Beijing, August 2010

ID#	Feature Name	Category	Feature Values
1	Consonant Inventories	Phonology (19)	{1:Large, 2:Small, 3:Moderately Small, 4:Moderately Large, 5:Average}
23	Locus of Marking in the Clause	Morphology (10)	{1:Head, 2:None, 3:Dependent, 4:Double, 5:Other}
30	Number of Genders	Nominal Categories (28)	{1:Three, 2:None, 3:Two, 4:Four, 5:Five or More}
58	Obligatory Possessive Inflection	Nominal Syntax (7)	{1:Absent, 2:Exists}
66	The Perfect	Verbal Categories (16)	{1:None, 2:Other, 3:From 'finish' or 'already', 4:From Possessive}
81	Order of Subject, Object and Verb	Word Order (17)	{1:SVO, 2:SOV, 3:No Dominant Order, 4:VSO, 5:VOS, 6:OVS, 7:OSV}
121	Comparative Constructions	Simple Clauses (24)	{1:Conjoined, 2:Locational, 3:Particle, 4:Exceed}
125	Purpose Clauses	Complex Sentences (7)	{1:Balanced/deranked, 2:Deranked, 3:Balanced}
138	Tea	Lexicon (10)	{1:Other, 2:Derived from Sinitic 'cha', 3:Derived from Chinese 'te'}
140	Question Particles in Sign Languages	Sign Languages (2)	{1:None, 2:One, 3:More than one}
142	Para-Linguistic Usages of Clicks	Other (2)	{1:Logical meanings, 2:Affective meanings, 3:Other or none}

Table 1: Sample features and their values used in the WALS database. There are eleven feature categories in WALS, one feature from each is given here. The numbers in parentheses in the 'Category' column are the total number of features in that category. Feature values are given with both the integers that represent them in the database and their description in the form {#:description}.

2 WALS

The WALS project consists of a database that catalogs linguistic features for over 2,556 languages in 208 language families, using 142 features in 11 different categories.[1] Table 1 shows a small sample of features, one feature from each category in WALS. Listed are the ID number for each example, the feature category, and the possible values for that feature.

WALS as a resource, however, is primarily designed for surveying the distribution of particular typological features worldwide, not comparing languages. The authors of WALS compiled their data from a wide array of primary sources, but these sources do not always cover the same sets of features or languages.

If we conceive of the WALS database as a two-dimensional matrix with languages along one dimension and features along the other, then only 16% of the cells in that matrix are filled. An empty cell in the matrix means the feature value for the (language, feature) pair is *not-specified (NS)*. Even well-studied languages could have many empty cells in WALS, and this kind of data sparsity presents serious problems to clustering algorithms that cannot handle unknown values. To address the data sparsity problem, we experiment with different pruning criteria to create a new matrix that is reasonably dense for our study.

[1]Our copy of the database was downloaded from `http://wals.info` in June of 2009 and appears to differ slightly from the statistics given on the website at the time of writing. Currently, the WALS website reports 2,650 languages, with 141 features in use.

2.1 Pruning Methods

Answering questions Q1–Q3 is difficult if there are too many empty cells in the data. Pruning the data to produce a smaller but denser subset can be done by one or more of the following methods.

Prune Languages by Minimum Features

Perhaps the most straightforward method of pruning is to eliminate languages that fail to contain some minimum number of features. Following Daumé (2009), we require languages to have a minimum of 25 features for the whole-world set, or 10 features for comparing across subfamilies. This eliminates many languages that simply do not have enough features to be adequately represented.

Prune Features by Minimum Coverage

The values for some features, such as those specific to sign languages, are provided only for a very small number of languages. Taking this into account, in addition to removing languages with a small number of features, it is also helpful to remove features that only cover a small portion of languages. Again we choose the thresholds selected by Daumé (2009) for pruning features that do not cover more than 10% of the selected languages in the whole-world set, and 25% in comparisons across subfamilies.

Use a Dense Language Family

Finally, using a well-studied family with a number of subfamilies can produce data sets with less sparsity. When clustering methods are used with this data, the groups correspond to subfamilies

Data Set	Min Features	Min Coverage	Grouped By	# Langs	# Groups	# Features	Density
Unpruned	0	0%	Family	2556	208	142	16.0%
Whole-World	25	10%	Family	735	121	139	39.7%
Indo-European	10	25%	Subfamily	87	10	64	44.9%
Sino-Tibetan	10	25%	Subfamily	96	14	64	38.6%

Table 2: Data sets and pruning options used for this paper. Density $= \frac{|Filled\ Cells|}{|Total\ Cells|} \cdot 100$

rather than families. In this study, we choose two families: Indo-European and Sino-Tibetan.

The resulting data sets after various methods of pruning can be seen in Table 2.

2.2 Features and Feature Values

Besides dealing with the sparsity of the features, the actual representation of the features in WALS needs to be taken into account. As can be seen in Table 1, features are represented with a range of discrete integer values. Some features, such as #58–Obligatory Possessive Inflection–are essentially binary features with values "`Absent`" or "`Exists`". Others, such as #1–Consonant Inventories–appear to be indices along some dimension related to size, ranging from small to large. Features such as these might conceivably be viewed as on a continuum where closer distances between values suggests closer relationship between languages.

Still other features, such as #81–Order of Subject, Object, and Verb–have multiple values but cannot be clearly be treated using distance measures. It's unclear how such a distance would vary between an SOV language and either VSO or VOS languages.

Binarization

Clustering algorithms use similarity functions, and some functions may simply check whether two languages have the same value for a feature. In these cases, no feature binarization is needed. If a clustering algorithm requires each data point (a language in this case) to be presented as a feature vector, features with more than two categorical values should be binarized. We simply treat a feature with k possible values as k binary features. There are other ways to binarize features. For instance, Daumé (2009) chose one feature value as the "canonical" value and grouped the other values into the second value (personal communication). We did not use this approach as it is not clear to us which values should be selected as the "canonical" ones.

3 Experimental Setup

To get a picture of how clustering methods compare to genetic groupings, we looked at three elements: cluster similarity, prediction capability, and feature selection.

3.1 Clustering

Our first experiment is designed to address question Q1: how do induced clusters compare to phylogenetic groupings?

Clustering Methods

For clustering, two clustering packages were used. First, we implemented the k-medoids algorithm, a partitional algorithm similar to k-means, but using median instead of mean distance for cluster centers (Estivill-Castro and Yang, 2000).

Second, we used a variety of methods from the CLUTO (Steinbach et al., 2000) clustering toolkit: repeated-bisection (`rb`), a k-means implementation (`direct`), an agglomerative algorithm (`agglo`) using UPGMA to produce hierarchical clusters, and `bagglo`, a variant of `agglo`, which biases the agglomerative algorithm using partitional clusters.

Similarity Measures

For similarity measures, we used CLUTO's default cosine similarity measure (`cos`), but also implemented another similarity measure `shared_overlap` designed to handle empty cells. Given two languages A and B, $shared\_overlap(A, B)$ is defined to be $\frac{\text{\# Of Features with Same Values}}{\text{\# Features Both Filled Out in WALS}}$. This measure can handle language pairs with many empty cells in WALS as it uses only features with cells

a is the number of language pairs found in the same set in both clusterings.
b is the number of language pairs found in different sets in C_1, and different sets in C_2.
c is the number of language pairs found in the same set in C_1, but in different sets in C_2.
d is the number of language pairs found in different sets in C_1, but the same set in C_2.

(a) Variables Used In Calculations

$$R(C_1, C_2) = \frac{a+b}{a+b+c+d}$$

(b) Rand Index

$$Precision(C_1, C_2) = \frac{a}{a+c}$$

(c) Cluster precision

$$Recall(C_1, C_2) = \frac{a}{a+d}$$

(d) Cluster recall

$$Fscore(C_1, C_2) = \frac{2 \cdot (Precision \cdot Recall)}{Precision + Recall}$$

(e) Cluster f-score

Figure 1: Formulas for calculating the Rand Index, cluster precision, recall, and f-score of two clusterings C_1 and C_2. C_1 is the system output, C_2 is the gold standard.

filled out for both languages, and calculates the percentage of features with the same values.

3.2 Clustering Performance Metrics

To measure clustering performance, we treat the genetic families specified in WALS as the gold standard, although we are not strictly aiming to recreate them.

Rand Index

The Rand Index (Rand, 1971) is one of the standard metrics for evaluating clustering results. It compares pairwise assignments of data points across two clusterings. For every pair of points there are four possibilities, as given in Figure 1. The Rand index is calculated by dividing the number of matching pairs ($a + b$) by the number of all pairs. This results in a number between 0 and 1 where 1 represents an identical clustering. Unfortunately, as noted by (Daumé and Marcu, 2005), the Rand Index tends to give disproportionately greater scores to clusterings with a greater number of clusters. For example, the Rand Index will always be 1.0 when each data point belongs to its own cluster. As a result, we have chosen to calculate metrics other than the Rand index: cluster precision, recall, and f-score.

Cluster Precision, Recall, and F-Score

Extending the notation in Figure 1, precision is defined as the proportion of same-set pairs in the target cluster C_1 that are correctly identified as being in the same set in the gold cluster C_2, while recall is the proportion of all same-set pairs in the gold cluster C_2 that are identified in the target cluster C_1. F-score is calculated as the usual harmonic mean of precision and recall. As it gives a more accurate representation of cluster similarity across varying amounts of clusters, we will report cluster similarity using cluster F-score.

3.3 Prediction Accuracy

Our second experiment was to answer the question posed in Q2: how do induced clusters and genetic families compare in predicting the values of features for languages in the same group?

To answer this question, we measure the accuracy of the prediction when both types of groups are used to predict the values of "empty" cells. We used 90% of the filled cells to build clusters, and then predicted the values of the remaining 10% of filled cells. The missing cells are filled with the value that occurs the most times among languages in the same group. If there are no other languages in the cluster, or the other languages have no values for this feature, then the cell is filled with the most common values for that feature across all languages in the dataset. Finally, the accuracy is calculated by comparing these predicted values with the actual values in the gold standard. We run 10-fold cross validation and report the average accuracy.

In addition to the prediction accuracy for each method of producing groupings, we calculate the baseline result where an empty cell is filled with the most frequent value for that feature across all the languages in the training data.

3.4 Determining Feature Stability

Finally, we look to answer Q3: what typological features tend to stay the same within related families? To find an answer, we look again to prediction accuracy. While prediction accuracy can be averaged across all features, it can also be broken down feature-by-feature to rank features according to how accurately they can be predicted by language families. Features that can be predicted with high accuracy implies that these features are more likely to remain *stable* within a language family than others.

Using prediction accuracies based on the genetic families, we rank features according to their accuracy and then perform clustering using the top features to determine if the cluster similarity to the genetic groups increases when using only the stable features.

4 Results & Analysis

4.1 Cluster Similarity

The graph in Figure 2(a) shows f-scores of clustering methods with the whole-world set. None achieve an f-score greater than 0.15, and most perform even worse when the number of clusters matches the number of genetic families or subfamilies. This indicates that the induced clusters based on typological features are very different from genetic groupings.

The question of similarity between these induced clusters and the genetic families is however a separate one from how those clusters perform in predicting typological feature values.

4.2 Prediction Accuracy

To determine the amount of similarity between languages within clusters, we instead look at prediction accuracy across clustering methods and the genetic groups. These scores are similar to those given in Daumé (2009), though not directly comparable due to small discrepancies in the size of the data set. As can be seen by the numbers in Table 3 and the graph in 2(b), despite the lack of similarity between clustering methods and the genetic groups, the clustering methods produce as good or better prediction accuracies. Furthermore, the `agglo` and `bagglo` hierarchical clustering methods which are favored for producing phylogenetically motivated clusters do indeed result in higher f-score similarity to the genetic clusters than the partitional `rb` and `direct` methods, but produce poorer prediction-accuracy results.

In fact, it is not surprising that some induced clusters outperform the genetic groupings in prediction accuracy, considering that clustering algorithms often want to maximize the similarity between languages in the same clusters. Now that we know similarity between languages does not necessarily mirror language family membership, the next question is what features tend to stay the same among languages in the same language families.

4.3 Feature Selection

Our final experiment was to examine the features in WALS themselves, and look for features that appear to vary the least within families, and act as better predictors of family membership.

In order to do this, we again looked at prediction accuracy information on a feature-by-feature basis. The results from this experiment are shown in Table 4, which gives a breakdown of how features rank both individually and by category.

Since this table is built upon genetic relationships, it is not surprising that the category for "Lexicon" appears to be the most reliably stable category. As noted in (McMahon, 1994), lexical cognates are often used as good evidence for determining a shared ancestry. We also find that word order is rather stable within a family.

We ran one further experiment where, using the `agglo` clustering method that provided clusters most similar to the genetic families previously, only features that showed accuracies above 50%. This eliminated 28 features, leaving 111 higher-scoring features for the whole-world set. Pruning the features to use only these selected for their stability within the genetic groupings yielded a very small increase in f-score similarity, as can be seen in Figure 3. Although this increase is small, it suggests that more advanced feature selection methods may be able to reveal language features that are more resistant to language contact and language change.

5 Error Analysis

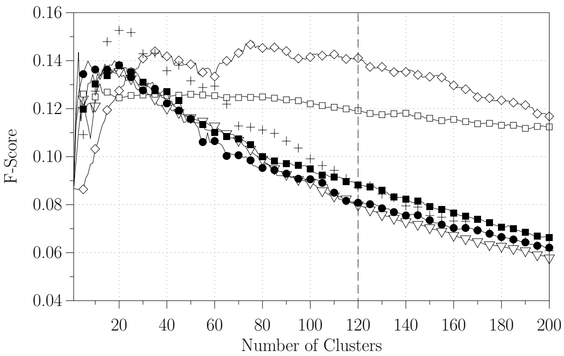
There are two main reasons for the differences between induced clusters and genetic groupings.

5.1 Language Similarity vs. Genetic Relatedness

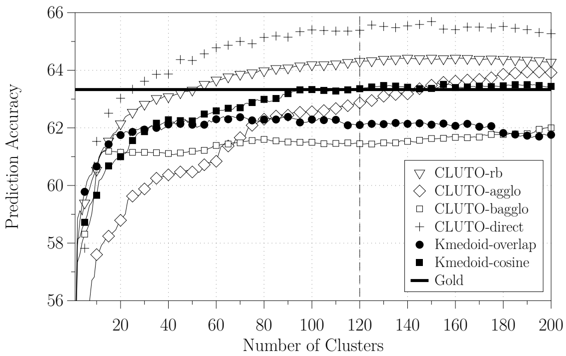
As mentioned before, language similarity and genetic relatedness are two different concepts. Simi-

	baseline	gold	rb	agglo	bagglo	direct	k-medoids with similarity overlap	k-medoids with cosine similarity
Whole-World-Set (121 Clusters)								
F-Score	0.087	–	0.080	**0.140**	0.119	0.089	0.081	0.088
Acc (%)	53.72	63.43	64.33	62.86	61.44	**65.47**	62.11	63.36
Indo-European Subset (10 Clusters)								
F-Score	0.319	–	0.365	0.377	**0.391**	0.355	0.352	0.331
Acc (%)	64.27	74.1	71.12	72.26	70.62	**74.13**	73.36	72.12
Sino-Tibetan Subset (14 Clusters)								
F-Score	0.305	–	0.224	**0.340**	0.333	0.220	0.285	0.251
Acc (%)	58.08	61.71	63.93	63.74	63.06	**65.31**	64.55	63.94

Table 3: Comparison of clustering algorithms when the number of clusters is set to the same number of genetic groupings. The highest number in each row is in boldface.



(a) F-scores of clustering results



(b) Prediction accuracy

Figure 2: Comparison of the performances of different clustering methods using the whole-world data set. The number of groups in the gold standard (i.e., genetic grouping) is shown as a vertical dashed line in 2(a) and 2(b), and the prediction accuracy of the gold standard as a horizontal solid line in 2(b).

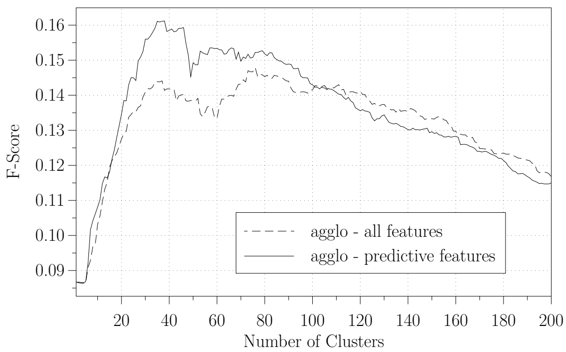


Figure 3: F-scores of the `agglo` clustering method when using all the features vs. only features whose prediction accuracy by the genetic grouping is higher than 50%.

lar languages might not be genetically related and dissimilar languages might be genetically related. An example is given in Table 5. Persian and English are both Indo-European languages, but look very different typologically; in contrast, Finnish and English are not genetically related but they look more similar typologically. While English and Persian are related, they have been diverging in geographically distant areas for thousands of years. Thus, the fact that English appears to share more features with a geographically closer Finnish is expected.

5.2 WALS as the Dataset

Perhaps the biggest challenge we encounter in this project has been the dataset itself. WALS has certain properties that complicate the task.

Data Sparsity and Shared Features

While the previous example shows unrelated languages can be quite similar typologically, our clustering methods put two closely related languages, Eastern and Western Armenian, into dif-

Breakdown by Feature Category		Breakdown By Feature: Top 10				Breakdown by Feature: Bottom 10			
Category	Accuracy	Feature	Acc	C	V	Feature	Acc	C	V
Whole-World Set									
Lexicon	75.0%	(136) M-T Pronouns	94.0%	230	3	(1) Consonant Inventories	32.6%	561	5
Word Order	68.6%	(18) Absence of Common Consonants	93.7%	565	6	(133) Number of Basic Color Categories	33.3%	119	7
Phonology	65.9%	(11) Front Rounded Vowels	91.1%	560	4	(23) Locus of Marking in the Clause	33.9%	236	5
Complex Sentences	64.0%	(73) The Optative	89.6%	319	2	(71) The Prohibitive	34.6%	495	4
Nominal Syntax	63.2%	(137) N-M Pronouns	87.9%	230	3	(22) Inflectional Synthesis of the Verb	35.1%	145	7
Verbal Categories	61.9%	(6) Uvular Consonants	85.0%	565	4	(56) Conjunctions and Universal Quantifiers	38.2%	116	3
Simple Clauses	60.5%	(130) Finger and Hand	84.4%	591	2	(117) Predicative Possession	39.4%	240	5
Nominal Categories	59.1%	(115) Negative Indefinite Pronouns	84.2%	206	4	(92) Position of Polar Question Particles	40.0%	775	6
Morphology	53.9%	(19) Presence of Uncommon Consonants	83.0%	565	7	(38) Indefinite Articles	40.4%	473	5
Other	41.3%	(58) Obligatory Possessive Inflection	81.4%	244	2	(50) Asymmetrical Case-Marking	40.7%	261	6
Indo-European Subset									
Lexicon	86.4%	(130) Finger and Hand	100.0%	35	2	(3) Consonant-Vowel Ratio	30.6%	31	5
Morphology	83.1%	(118) Predicative Adjectives	100.0%	29	3	(92) Position of Polar Question Particles	34.6%	47	6
Word Order	79.6%	(18) Absence of Common Consonants	100.0%	31	6	(78) Coding of Evidentiality	36.0%	23	6
Simple Clauses	76.6%	(107) Passive Constructions	100.0%	19	2	(1) Consonant Inventories	42.4%	31	5
Nominal Categories	70.4%	(88) Order of Demonstrative and Noun	97.2%	66	6	(2) Vowel Quality Inventories	44.4%	31	3
Phonology	66.7%	(89) Order of Numeral and Noun	95.7%	64	4	(84) Order of Object, Oblique, and Verb	47.8%	20	6
Verbal Categories	62.1%	(27) Reduplication	95.2%	20	3	(16) Weight Factors in Weight-Sensitive Stress Systems	51.1%	53	7
		(7) Glottalized Consonants	93.9%	31	8	(70) The Morphological Imperative	55.3%	53	5
		(93) Position of Interrogative Phrases in Content Questions	93.9%	44	3	(44) Gender Distinctions in Independent Personal Pronouns	56.5%	19	6
		(5) Voicing and Gaps in Plosive Systems	93.8%	31	5	(37) Definite Articles	59.2%	46	5
Sino-Tibetan Subset									
Lexicon	100.0%	(130) Finger and Hand	100.0%	8	2	(77) Semantic Distinctions of Evidentiality	9.1%	18	3
Word Order	67.7%	(82) Order of Subject and Verb	100.0%	99	3	(78) Coding of Evidentiality	17.7%	18	6
Morphology	63.8%	(119) Nominal and Locational Predication	100.0%	13	2	(4) Voicing in Plosives and Fricatives	20.7%	26	4
Simple Clauses	60.9%	(86) Order of Genitive and Noun	100.0%	73	3	(1) Consonant Inventories	22.2%	26	5
Verbal Categories	60.7%	(129) Hand and Arm	100.0%	8	2	(14) Fixed Stress Locations	25.0%	4	7
Nominal Categories	55.8%	(18) Absence of Common Consonants	100.0%	26	6	(15) Weight-Sensitive Stress	25.0%	4	8
Phonology	50.7%	(93) Pos. of Interr. Phrases in Content Q's	100.0%	79	3	(38) Indefinite Articles	31.7%	36	5
		(85) Order of Adposition and Noun Phrase	97.5%	79	5	(120) Zero Copula for Predicate Nominals	37.5%	13	2
		(95) Relationship b/t Object and Verb and Adposition and Noun Phrase	96.3%	76	5	(2) Vowel Quality Inventories	42.9%	26	3
		(48) Person Marking on Adpositions	93.3%	14	4	(3) Consonant-Vowel Ratio	46.7%	26	5

Table 4: Prediction accuracy figures derived from genetic groupings for each dataset and broken down by WALS feature category and feature. Ordering is by descending accuracy for the top 10 features, and by increasing accuracy for the bottom 10 features. The 'C' and 'V' columns give the number of languages in the set that a feature appears in, and the number of possible values for that feature, respectively.

ferent clusters. A quick review shows that the reason for this mistake is due to a lack of shared features in WALS. Table 6 shows that very few features are specified for both languages. The data sparsity problem and the distribution of empty cells adversely affect clustering results.

Notice that in this example, the features whose values are filled for both languages actually have identical feature values. While using shared overlap as a similarity measure can capture the similarity between these two languages, this measure biases clustering toward features with fewer cells filled out. The only way out of errors like this, it seems, is to obtain more data.

There are a few other typological databases that might be drawn upon to define a more complete set of data: PHOIBLE, (Moran and Wright, 2009), ODIN (Lewis, 2006), and the AUTOTYP database (Nichols and Bickel, 2009). Using these databases to fill in the gaps in data may be the only way to fully address these issues.

The Feature Set in WALS

The features in WALS are not systematically chosen for full typological coverage; rather, the contributors to WALS decide what features they want to work on based on their expertise. Also, some features in WALS overlap; for example, one WALS feature looks at the order between subject, verb, and object, and another feature checks the order between verb and object. As a result, the feature set in WALS might not be a good representative of the properties of the languages covered in the database.

6 Conclusion & Further Work

By comparing clusters derived from typological features to genetic groups in the world's languages, we have found two interesting results. First, the induced clusters look very different from genetic grouping and this is partly due to the design of WALS. Second, despite the differences, induced clusters show similar, or even greater levels

ID: Feature Name	English	Finnish	Persian
2: Vowel Quality Inventories	**Large (7-14)**	**Large (7-14)**	Average (5-6)
6: Uvular Consonants	**None**	**None**	Uvular stops only
11: Front Rounded Vowels	**None**	High and Mid	**None**
27: Reduplication	**No productive reduplication**	**No productive reduplication**	Productive full and partial reduplication
37: Definite Articles	Definite word distinct from demonstrative	No definite or indefinite article	No definite, but indefinite article
53: Ordinal Numerals	**First, second, three-th**	**First, second, three-th**	First/one-th, two-th, three-th
81: Order of Subject, Object and Verb	**SVO**	**SVO**	SOV
85: Order of Adposition and Noun Phrase	**Prepositions**	Postpositions	**Prepositions**
87: Order of Adjective and Noun	**Adjective-Noun**	**Adjective-Noun**	Noun-Adjective
124: 'Want' Complement Subjects	**Subject left implicit**	**Subject left implicit**	Subject expressed overtly
Number of Features	139	135	128
Cosine Similarity to Eng	1.00	0.56	0.42
Shared Overlap with Eng	1.00	0.56	0.44

Table 5: A selection of ten features from English, Finnish, and Persian. Same feature values in each row are in boldface. Despite the genetic relation between English and Persian, similarity metrics place English closer to Finnish than Persian.

ID#	Feature Name	Armenian (Eastern)	Armenian (Western)
1	Consonant Inventories	Small	–
27	Reduplication	**Full Reduplication Only**	**Full Reduplication Only**
33	Coding of Nominal Plurality	–	Plural suffix
48	Person Marking on Adj.	None	–
81	Order of Subj. Obj., and V	–	SOV
86	Order of Adposition and Noun Phrase	**Postpositions**	**Postpositions**
100	Alignment of Verbal Person Marking	Accusative	–
129	Hand and Arm	–	Identical
	Number of Features	85	33
	Cosine Similarity	0.22	
	Shared Overlap	1.00	

Table 6: Comparison of features between Eastern and Western Armenian. Same feature values in each row are in boldface. Empty cells are shown as '–'.

of typological similarity than genetic grouping as indicated by the prediction accuracy.

While these initial findings are interesting, using WALS as a dataset for this purpose leaves a lot to be desired. Subsequent work that supplements the typological data in WALS with the databases mentioned in §5.2 would help alleviate the data sparsity and feature selection problems.

Another useful follow-up would be to perform application-oriented evaluations. For instance, evaluating the performance of syntactic projection methods between languages determined to have similar syntactic patterns, or using similar morphological induction techniques on morphologically similar languages. With the development of large typological databases such as WALS, we hope to see more studies that take advantage of resources for resource-rich languages when developing tools for typologically similar, but resource-poor languages.

Acknowledgment This work is supported by the National Science Foundation Grant BCS-0748919. We would also like to thank Emily Bender, Tim Baldwin, and three anonymous reviewers for helpful comments.

References

Daumé, III, Hal and Daniel Marcu. 2005. A Bayesian Model for Supervised Clustering with the Dirichlet Process Prior. *Journal of Machine Learning Research*, 6:1551–1577.

Daumé, III, Hal. 2009. Non-Parametric Bayesian Areal Linguistics. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL)*, pages 593–601, Boulder, Colorado, June.

Estivill-Castro, Vladimir and Jianhua Yang. 2000. A fast and robust general purpose clustering algorithm. In *Proc. of Pacific Rim International Conference on Artificial Intelligence*, pages 208–218. Springer.

Hana, Jiri, Anna Feldman, and Chris Brew. 2004. A Resource-light Approach to Russian Morphology: Tagging Russian using Czech resources. In *Proceedings of EMNLP 2004*, Barcelona, Spain.

Haspelmath, Martin, Matthew S. Dryer, David Gil, and Bernard Comrie. 2005. *The World Atlas of Language Structures*. Oxford University Press, Oxford, England.

Lewis, William D. 2006. ODIN: A Model for Adapting and Enriching Legacy Infrastructure. In *Proceedings of the e-Humanities Workshop, held in cooperation with e-Science 2006: 2nd IEEE International Conference on e-Science and Grid Computing*, Amsterdam.

McMahon, April M. S. 1994. *Understanding language change*. Cambridge University Press, Cambridge; New York, NY, USA.

Moran, Steven and Richard Wright. 2009. Phonetics Information Base and Lexicon (PHOIBLE). Online: `http://phoible.org`.

Nichols, Johanna and Balthasar Bickel. 2009. The AUTOTYP genealogy and geography database: 2009 release. `http://www.uni-leipzig.de/~autotyp`.

Rand, William M. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850.

Steinbach, Michael, George Karypis, and Vipin Kumar. 2000. A comparison of document clustering techniques. In *Proceedings of Workshop at KDD 2000 on Text Mining*.

Xia, Fei and William D. Lewis. 2007. Multilingual structural projection across interlinear text. In *Proc. of the Conference on Human Language Technologies (HLT/NAACL 2007)*, pages 452–459, Rochester, New York.

Yarowsky, David and Grace Ngai. 2001. Inducing multilingual pos taggers and np bracketers via robust projection across aligned corpora. In *Proc. of the Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies (NAACL-2001)*, pages 1–8, Morristown, NJ, USA.

Better Arabic Parsing: Baselines, Evaluations, and Analysis

Spence Green and Christopher D. Manning
Computer Science Department, Stanford University
{spenceg,manning}@stanford.edu

Abstract

In this paper, we offer broad insight into the underperformance of Arabic constituency parsing by analyzing the interplay of linguistic phenomena, annotation choices, and model design. First, we identify sources of syntactic ambiguity understudied in the existing parsing literature. Second, we show that although the Penn Arabic Treebank is similar to other treebanks in gross statistical terms, annotation consistency remains problematic. Third, we develop a human interpretable grammar that is competitive with a latent variable PCFG. Fourth, we show how to build better models for three different parsers. Finally, we show that in application settings, the absence of gold segmentation lowers parsing performance by 2–5% F1.

1 Introduction

It is well-known that constituency parsing models designed for English often do not generalize easily to other languages and treebanks.[1] Explanations for this phenomenon have included the relative informativeness of lexicalization (Dubey and Keller, 2003; Arun and Keller, 2005), insensitivity to morphology (Cowan and Collins, 2005; Tsarfaty and Sima'an, 2008), and the effect of variable word order (Collins et al., 1999). Certainly these linguistic factors increase the difficulty of syntactic disambiguation. Less frequently studied is the interplay among language, annotation choices, and parsing model design (Levy and Manning, 2003; Kübler, 2005).

To investigate the influence of these factors, we analyze Modern Standard Arabic (henceforth MSA, or simply "Arabic") because of the unusual opportunity it presents for comparison to English parsing results. The Penn Arabic Treebank (ATB) syntactic guidelines (Maamouri et al., 2004) were purposefully borrowed without major modification from English (Marcus et al., 1993). Further, Maamouri and Bies (2004) argued that the English guidelines generalize well to other languages. But Arabic contains a variety of linguistic phenomena unseen in English. Crucially, the conventional orthographic form of MSA text is *unvocalized*, a property that results in a deficient graphical representation. For humans, this characteristic can impede the acquisition of literacy. How do additional ambiguities caused by devocalization affect statistical learning? How should the absence of vowels and syntactic markers influence annotation choices and grammar development? Motivated by these questions, we significantly raise baselines for three existing parsing models through better grammar engineering.

Our analysis begins with a description of syntactic ambiguity in unvocalized MSA text (§2). Next we show that the ATB is similar to other treebanks in gross statistical terms, but that annotation consistency remains low relative to English (§3). We then use linguistic and annotation insights to develop a manually annotated grammar for Arabic (§4). To facilitate comparison with previous work, we exhaustively evaluate this grammar and two other parsing models when gold segmentation is assumed (§5). Finally, we provide a realistic evaluation in which segmentation is performed both in a pipeline and jointly with parsing (§6). We quantify error categories in both evaluation settings. To our knowledge, ours is the first analysis of this kind for Arabic parsing.

[1]The apparent difficulty of adapting constituency models to non-configurational languages has been one motivation for dependency representations (Hajič and Zemánek, 2004; Habash and Roth, 2009).

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 394–402,
Beijing, August 2010

2 Syntactic Ambiguity in Arabic

Arabic is a morphologically rich language with a root-and-pattern system similar to other Semitic languages. The basic word order is VSO, but SVO, VOS, and VO configurations are also possible.[2] Nouns and verbs are created by selecting a consonantal root (usually triliteral or quadriliteral), which bears the semantic core, and adding affixes and diacritics. Particles are uninflected. Diacritics can also be used to specify grammatical relations such as case and gender. But diacritics are not present in unvocalized text, which is the standard form of, e.g., news media documents.[3]

Let us consider an example of ambiguity caused by devocalization. Table 1 shows four words whose unvocalized surface forms ان *an* are indistinguishable. Whereas Arabic linguistic theory assigns (1) and (2) to the class of pseudo verbs إن وأخواتها *inna and her sisters* since they can be inflected, the ATB conventions treat (2) as a complementizer, which means that it must be the head of SBAR. Because these two words have identical complements, syntax rules are typically unhelpful for distinguishing between them. This is especially true in the case of quotations—which are common in the ATB—where (1) will follow a verb like (2) (Figure 1).

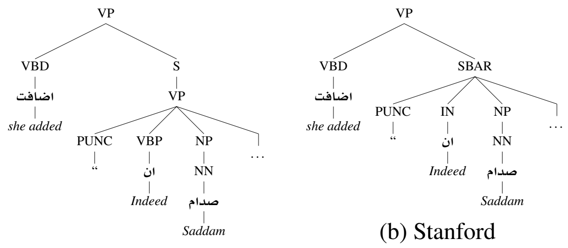
Even with vocalization, there are linguistic categories that are difficult to identify without semantic clues. Two common cases are the attributive adjective and the process nominal المصدار *maSdar*, which can have a verbal reading.[4] Attributive adjectives are hard because they are orthographically identical to nominals; they are inflected for gender, number, case, and definiteness. Moreover, they are used as substantives much more frequently than is done in English.

Process nominals name the action of the transitive or ditransitive verb from which they derive. The verbal reading arises when the *maSdar* has an NP argument which, in vocalized text, is marked in the accusative case. When the *maSdar* lacks a determiner, the constituent as a whole resembles the ubiquitous annexation construct الإضافة *iDafa*. Gabbard and Kulick (2008) show that there is significant attachment ambiguity associated with *iDafa*, which occurs in 84.3% of the trees in our development set. Figure 4 shows a constituent headed by a process nominal with an embedded adjective phrase. All three models evaluated in this paper incorrectly analyze the constituent as *iDafa*; none of the models attach the attributive adjectives properly.

For parsing, the most challenging form of ambiguity occurs at the discourse level. A defining characteristic of MSA is the prevalence of *discourse markers* to connect and subordinate words and phrases (Ryding, 2005). Instead of offsetting new topics with punctuation, writers of MSA insert connectives such as و *wa* and ف *fa* to link new elements to both preceding clauses and the text as a whole. As a result, Arabic sentences are usually long relative to English, especially after

	Word	Head Of	Complement	POS
1	إنّ *inna* "Indeed, truly"	VP	Noun	VBP
2	أنّ *anna* "That"	SBAR	Noun	IN
3	إن *in* "If"	SBAR	Verb	IN
4	أن *an* "to"	SBAR	Verb	IN

Table 1: Diacritized particles and pseudo-verbs that, after orthographic normalization, have the equivalent surface form ان *an*. The distinctions in the ATB are linguistically justified, but complicate parsing. Table 8a shows that the best model recovers SBAR at only 71.0% F1.



(a) Reference (b) Stanford

Figure 1: The Stanford parser (Klein and Manning, 2002) is unable to recover the verbal reading of the unvocalized surface form ان *an* (Table 1).

[2]Unlike machine translation, constituency parsing is not significantly affected by variable word order. However, when grammatical relations like subject and object are evaluated, parsing performance drops considerably (Green et al., 2009). In particular, the decision to represent arguments in verb-initial clauses as VP internal makes VSO and VOS configurations difficult to distinguish. Topicalization of NP subjects in SVO configurations causes confusion with VO (pro-drop).

[3]Techniques for automatic vocalization have been studied (Zitouni et al., 2006; Habash and Rambow, 2007). However, the data sparsity induced by vocalization makes it difficult to train statistical models on corpora of the size of the ATB, so vocalizing and then parsing may well not help performance.

[4]Traditional Arabic linguistic theory treats both of these types as subcategories of noun الاسم.

Length	English (WSJ)	Arabic (ATB)
≤ 20	41.9%	33.7%
≤ 40	**92.4%**	73.2%
≤ 63	99.7%	**92.6%**
≤ 70	99.9%	94.9%

Table 2: Frequency distribution for sentence lengths in the WSJ (sections 2–23) and the ATB (p1–3). English parsing evaluations usually report results on sentences up to length 40. Arabic sentences of up to length 63 would need to be evaluated to account for the same fraction of the data. We propose a limit of 70 words for Arabic parsing evaluations.

	Part of Speech	Tag	Freq.
و *wa* "and"	conjunction	CC	4256
	preposition	IN	6
	abbreviation	NN	6
ف *fa* "so, then"	conjunction	CC	160
	connective particle	RP	67
	abbreviation	NN	22
	response conditioning particle	RP	11
	subordinating conjunction	IN	3

Table 3: Dev set frequencies for the two most significant discourse markers in Arabic are skewed toward analysis as a conjunction.

segmentation (Table 2). The ATB gives several different analyses to these words to indicate different types of coordination. But it conflates the coordinating and discourse separator functions of *wa* (واو العطف) into one analysis: conjunction (Table 3). A better approach would be to distinguish between these cases, possibly by drawing on the vast linguistic work on Arabic connectives (Al-Batal, 1990). We show that noun-noun vs. discourse-level coordination ambiguity in Arabic is a significant source of parsing errors (Table 8c).

3 Treebank Comparison

3.1 Gross Statistics

Linguistic intuitions like those in the previous section inform language-specific annotation choices. The resulting structural differences between treebanks can account for relative differences in parsing performance. We compared the ATB[5] to treebanks for Chinese (CTB6), German (Negra), and English (WSJ) (Table 4). The ATB is disadvantaged by having fewer trees with longer average

	ATB	CTB6	Negra	WSJ
Trees	23449	28278	20602	**43948**
Word Typess	40972	45245	51272	**46348**
Tokens	738654	782541	355096	**1046829**
Tags	32	34	**499**	45
Phrasal Cats	22	26	**325**	27
Test OOV	16.8%	22.2%	**30.5%**	13.2%
Per Sentence				
Depth (μ / σ^2)	3.87 / 0.74	**5.01 / 1.44**	3.58 / 0.89	4.18 / 0.74
Breadth (μ / σ^2)	**14.6 / 7.31**	10.2 / 4.44	7.50 / 4.56	12.1 / 4.65
Length (μ / σ^2)	**31.5 / 22.0**	27.7 / 18.9	17.2 / 10.9	23.8 / 11.2
Constituents (μ)	**32.8**	32.5	8.29	19.6
μ Const. / μ Length	1.04	**1.18**	0.482	0.820

Table 4: Gross statistics for several different treebanks. Test set OOV rate is computed using the following splits: ATB (Chiang et al., 2006); CTB6 (Huang and Harper, 2009); Negra (Dubey and Keller, 2003); English, sections 2-21 (train) and section 23 (test).

yields.[6] But to its great advantage, it has a high ratio of non-terminals/terminals (μ Constituents / μ Length). Evalb, the standard parsing metric, is biased *toward* such corpora (Sampson and Babarczy, 2003). Also surprising is the low test set OOV rate given the possibility of morphological variation in Arabic. In general, several gross corpus statistics favor the ATB, so other factors must contribute to parsing underperformance.

3.2 Inter-annotator Agreement

Annotation consistency is important in any supervised learning task. In the initial release of the ATB, inter-annotator agreement was inferior to other LDC treebanks (Maamouri et al., 2008). To improve agreement during the revision process, a dual-blind evaluation was performed in which 10% of the data was annotated by independent teams. Maamouri et al. (2008) reported agreement between the teams (measured with Evalb) at 93.8% F1, the level of the CTB. But Rehbein and van Genabith (2007) showed that Evalb should not be used as an indication of real difference—or similarity—between treebanks.

Instead, we extend the *variation n-gram* method of Dickinson (2005) to compare annotation error rates in the WSJ and ATB. For a corpus C, let M be the set of tuples $\langle n, l \rangle$, where n is an n-gram with bracketing label l. If any n appears

[5]LDC A-E catalog numbers: LDC2008E61 (ATBp1v4), LDC2008E62 (ATBp2v3), and LDC2008E22 (ATBp3v3.1). We map the ATB morphological analyses to the shortened "Bies" tags for all experiments.

[6]Generative parsing performance is known to deteriorate with sentence length. As a result, Habash et al. (2006) developed a technique for splitting and chunking long sentences. In application settings, this may be a profitable strategy.

	Corpus		Sample	Error %	
	Trees	Nuclei	n-grams	Type	n-gram
WSJ 2–23	43948	25041	746	12.0%	**2.10%**
ATB	23449	20292	2100	**37.0%**	1.76%

Table 5: Evaluation of 100 randomly sampled variation nuclei types. The samples from each corpus were independently evaluated. The ATB has a much higher fraction of nuclei per tree, and a higher type-level error rate.

in a corpus position without a bracketing label, then we also add $\langle n, \texttt{NIL}\rangle$ to M. We call the set of unique n-grams with multiple labels in M the *variation nuclei* of C.

Bracketing variation can result from either annotation errors or linguistic ambiguity. Human evaluation is one way to distinguish between the two cases. Following Dickinson (2005), we randomly sampled 100 variation nuclei from each corpus and evaluated each sample for the presence of an annotation error. The human evaluators were a non-native, fluent Arabic speaker (the first author) for the ATB and a native English speaker for the WSJ.[7]

Table 5 shows type- and token-level error rates for each corpus. The 95% confidence intervals for type-level errors are (5580, 9440) for the ATB and (1400, 4610) for the WSJ. The results clearly indicate increased variation in the ATB relative to the WSJ, but care should be taken in assessing the magnitude of the difference. On the one hand, the type-level error rate is not calibrated for the number of n-grams in the sample. At the same time, the n-gram error rate is sensitive to samples with extreme n-gram counts. For example, one of the ATB samples was the determiner ذلك *dhalik* "that." The sample occurred in 1507 corpus positions, and we found that the annotations were consistent. If we remove this sample from the evaluation, then the ATB type-level error rises to only 37.4% while the n-gram error rate increases to 6.24%. The number of ATB n-grams also falls below the WSJ sample size as the largest WSJ sample appeared in only 162 corpus positions.

[7]Unlike Dickinson (2005), we strip traces and only consider POS tags when pre-terminals are the only intervening nodes between the nucleus and its bracketing (e.g., unaries, base NPs). Since our objective is to compare distributions of bracketing discrepancies, we do not use heuristics to prune the set of nuclei.

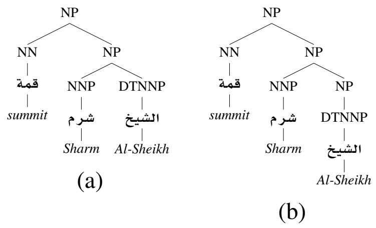


Figure 2: An ATB sample from the human evaluation. The ATB annotation guidelines specify that proper nouns should be specified with a flat NP (a). But the city name *Sharm Al-Sheikh* is also *iDafa*, hence the possibility for the incorrect annotation in (b).

4 Grammar Development

We can use the preceding linguistic and annotation insights to build a manually annotated Arabic grammar in the manner of Klein and Manning (2003). Manual annotation results in human interpretable grammars that can inform future treebank annotation decisions. A simple lexicalized PCFG with second order Markovization gives relatively poor performance: 75.95% F1 on the test set.[8] But this figure is surprisingly competitive with a recent state-of-the-art baseline (Table 7).

In our grammar, features are realized as annotations to basic category labels. We start with noun features since written Arabic contains a very high proportion of NPs. **genitiveMark** indicates recursive NPs with a indefinite nominal left daughter and an NP right daughter. This is the form of recursive levels in *iDafa* constructs. We also add an annotation for one-level *iDafa* (**oneLevelIdafa**) constructs since they make up more than 75% of the *iDafa* NPs in the ATB (Gabbard and Kulick, 2008). For all other recursive NPs, we add a common annotation to the POS tag of the head (**recursiveNPHead**).

Base NPs are the other significant category of nominal phrases. **markBaseNP** indicates these non-recursive nominal phrases. This feature includes named entities, which the ATB marks with a flat NP node dominating an arbitrary number of NNP pre-terminal daughters (Figure 2).

For verbs we add two features. First we mark any node that dominates (at any level) a verb

[8]We use head-finding rules specified by a native speaker of Arabic. This PCFG is incorporated into the Stanford Parser, a factored model that chooses a 1-best parse from the product of constituency and dependency parses.

Feature	States	Tags	F1	Indiv. ΔF1
—	3208	33	76.86	—
recursiveNPHead	3287	38	77.46	+0.60
genitiveMark	3471	38	77.88	+0.42
splitPUNC	4221	47	77.98	+0.10
markContainsVerb	5766	47	79.16	+1.18
markBaseNP	6586	47	79.5	+0.34
markOneLevelIdafa	7202	47	79.83	+0.33
splitIN	7595	94	80.48	+0.65
containsSVO	9188	94	80.66	+0.18
splitCC	9492	124	80.87	+0.21
markFem	10049	141	80.95	+0.08

Table 6: Incremental dev set results for the manually annotated grammar (sentences of length ≤ 70).

phrase (**markContainsVerb**). This feature has a linguistic justification. Historically, Arabic grammar has identified two sentences types: those that begin with a nominal (الجملة الاسمية), and those that begin with a verb (الجملة الفعلية). But foreign learners are often surprised by the verbless predications that are frequently used in Arabic. Although these are technically nominal, they have become known as "equational" sentences. **markContainsVerb** is especially effective for distinguishing root S nodes of equational sentences. We also mark all nodes that dominate an SVO configuration (**containsSVO**). In MSA, SVO usually appears in non-matrix clauses.

Lexicalizing several POS tags improves performance. **splitIN** captures the verb/preposition idioms that are widespread in Arabic. Although this feature helps, we encounter one consequence of variable word order. Unlike the WSJ corpus which has a high frequency of rules like VP $\rightarrow$ VB PP, Arabic verb phrases usually have lexicalized intervening nodes (e.g., NP subjects and direct objects). For example, we might have VP $\rightarrow$ VB NP PP, where the NP is the subject. This annotation choice weakens **splitIN**.

The ATB gives all punctuation a single tag. For parsing, this is a mistake, especially in the case of interrogatives. **splitPUNC** restores the convention of the WSJ. We also mark all tags that dominate a word with the feminine ending ة *taa marbuuTa* (**markFeminine**).

To differentiate between the coordinating and discourse separator functions of conjunctions (Table 3), we mark each CC with the label of its right sister (**splitCC**). The intuition here is that the role of a discourse marker can usually be determined by the category of the word that follows it. Because conjunctions are elevated in the parse trees when they separate recursive constituents, we choose the right sister instead of the category of the next word. We create equivalence classes for verb, noun, and adjective POS categories.

5 Standard Parsing Experiments

We compare the manually annotated grammar, which we incorporate into the Stanford parser, to both the Berkeley (Petrov et al., 2006) and Bikel (Bikel, 2004) parsers. All experiments use ATB parts 1–3 divided according to the canonical split suggested by Chiang et al. (2006). Preprocessing the raw trees improves parsing performance considerably.[9] We first discard all trees dominated by X, which indicates errors and non-linguistic text. At the phrasal level, we remove all function tags and traces. We also collapse unary chains with identical basic categories like NP $\rightarrow$ NP. The preterminal morphological analyses are mapped to the shortened "Bies" tags provided with the treebank. Finally, we add "DT" to the tags for definite nouns and adjectives (Kulick et al., 2006).

The orthographic normalization strategy we use is simple.[10] In addition to removing all diacritics, we strip instances of *taTweel* تطويل, collapse variants of *alif* ا to bare *alif*,[11] and map Arabic punctuation characters to their Latin equivalents. We retain segmentation markers—which are consistent only in the vocalized section of the treebank—to differentiate between e.g. هم "they" and هم+ "their." Because we use the vocalized section, we must remove null pronoun markers.

In Table 7 we give results for several evaluation metrics. Evalb is a Java re-implementation of the standard labeled precision/recall metric.[12]

[9] Both the corpus split and pre-processing code are available at http://nlp.stanford.edu/projects/arabic.shtml.

[10] Other orthographic normalization schemes have been suggested for Arabic (Habash and Sadat, 2006), but we observe negligible parsing performance differences between these and the simple scheme used in this evaluation.

[11] *taTweel* (ـ) is an elongation character used in Arabic script to justify text. It has no syntactic function. Variants of *alif* are inconsistently used in Arabic texts. For *alif* with *hamza*, normalization can be seen as another level of devocalization.

[12] For English, our Evalb implementation is identical to the most recent reference (EVALB20080701). For Arabic we

Model	System	Length	Leaf Ancestor			Evalb			Tag
			Corpus	Sent	Exact	LP	LR	F1	%
Stanford (v1.6.3)	Baseline	70	0.791	0.825	**358**	80.37	79.36	79.86	95.58
		all	0.773	0.818	358	78.92	77.72	78.32	95.49
	GoldPOS	70	0.802	0.836	452	81.07	80.27	80.67	99.95
Bikel (v1.2)	Baseline (Self-tag)	70	0.770	0.801	278	77.92	76.00	76.95	94.64
		all	0.752	0.794	278	76.96	75.01	75.97	94.63
	Baseline (Pre-tag)	70	0.771	0.804	295	78.35	76.72	77.52	**95.68**
		all	0.752	0.796	295	77.31	75.64	76.47	95.68
	GoldPOS	70	0.775	0.808	309	78.83	77.18	77.99	96.60
Berkeley (Sep. 09)	(Petrov, 2009)	all	—	—	—	76.40	75.30	75.85	—
	Baseline	70	**0.809**	**0.839**	335	**82.32**	**81.63**	**81.97**	95.07
		all	0.796	0.834	336	81.43	80.73	81.08	95.02
	GoldPOS	70	0.831	0.859	496	84.37	84.21	84.29	99.87

Table 7: Test set results. Maamouri et al. (2009b) evaluated the Bikel parser using the same ATB split, but only reported dev set results with gold POS tags for sentences of length ≤ 40. The Bikel GoldPOS configuration only supplies the gold POS tags; it does not force the parser to use them. We are unaware of prior results for the Stanford parser.

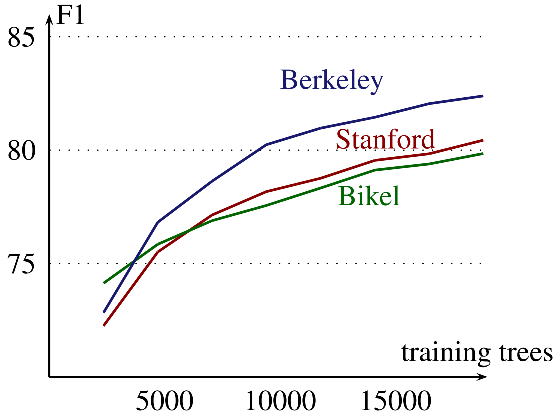


Figure 3: Dev set learning curves for sentence lengths ≤ 70. All three curves remain steep at the maximum training set size of 18818 trees.

The Leaf Ancestor metric measures the cost of transforming guess trees to the reference (Sampson and Babarczy, 2003). It was developed in response to the non-terminal/terminal bias of Evalb, but Clegg and Shepherd (2005) showed that it is also a valuable diagnostic tool for trees with complex deep structures such as those found in the ATB. For each terminal, the Leaf Ancestor metric extracts the shortest path to the root. It then computes a normalized Levenshtein edit distance between the extracted chain and the reference. The range of the score is between 0 and 1 (higher is better). We report micro-averaged (whole corpus) and macro-averaged (per sentence) scores along with the number of exactly matching guess trees.

add a constraint on the removal of punctuation, which has a single tag (PUNC) in the ATB. Tokens tagged as PUNC are not discarded unless they consist entirely of punctuation.

5.1 Parsing Models

The Stanford parser includes both the manually annotated grammar (§4) and an Arabic unknown word model with the following lexical features:

1. Presence of the determiner ال *Al*
2. Contains digits
3. Ends with the feminine affix ة *p*
4. Various verbal (e.g., وا, ت) and adjectival suffixes (e.g., ية)

Other notable parameters are second order vertical Markovization and marking of unary rules.

Modifying the Berkeley parser for Arabic is straightforward. After adding a ROOT node to all trees, we train a grammar using six split-and-merge cycles and no Markovization. We use the default inference parameters.

Because the Bikel parser has been parameterized for Arabic by the LDC, we do not change the default model settings. However, when we pre-tag the input—as is recommended for English—we notice a 0.57% F1 improvement. We use the log-linear tagger of Toutanova et al. (2003), which gives 96.8% accuracy on the test set.

5.2 Discussion

The Berkeley parser gives state-of-the-art performance for all metrics. Our baseline for all sentence lengths is 5.23% F1 higher than the best previous result. The difference is due to more careful

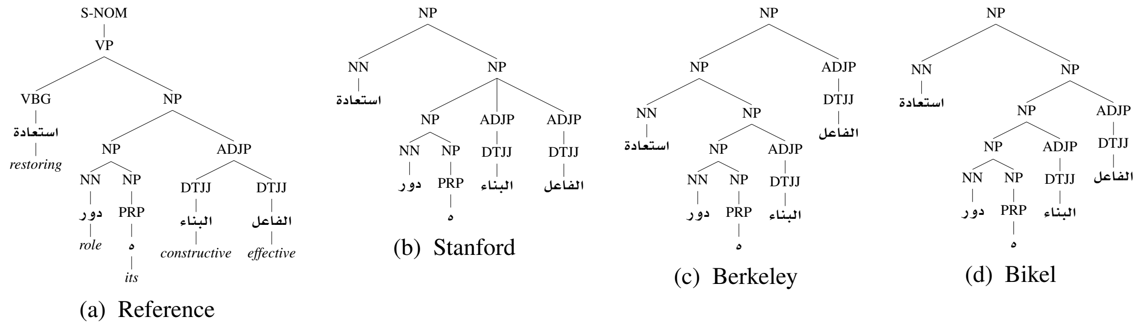


Figure 4: The constituent *Restoring of its constructive and effective role* parsed by the three different models (gold segmentation). The ATB annotation distinguishes between verbal and nominal readings of *maSdar* process nominals. Like verbs, *maSdar* takes arguments and assigns case to its objects, whereas it also demonstrates nominal characteristics by, e.g., taking determiners and heading *iDafa* (Fassi Fehri, 1993). In the ATB, استعادة *asta'adah* is tagged 48 times as a noun and 9 times as verbal noun. Consequently, all three parsers prefer the nominal reading. Table 8b shows that verbal nouns are the hardest pre-terminal categories to identify. None of the models attach the attributive adjectives correctly.

pre-processing. However, the learning curves in Figure 3 show that the Berkeley parser does not exceed our manual grammar by as wide a margin as has been shown for other languages (Petrov, 2009). Moreover, the Stanford parser achieves the most exact Leaf Ancestor matches and tagging accuracy that is only 0.1% below the Bikel model, which uses pre-tagged input.

In Figure 4 we show an example of variation between the parsing models. We include a list of per-category results for selected phrasal labels, POS tags, and dependencies in Table 8. The errors shown are from the Berkeley parser output, but they are representative of the other two parsing models.

6 Joint Segmentation and Parsing

Although the segmentation requirements for Arabic are not as extreme as those for Chinese, Arabic is written with certain cliticized prepositions, pronouns, and connectives connected to adjacent words. Since these are distinct syntactic units, they are typically segmented. The ATB segmentation scheme is one of many alternatives. Until now, all evaluations of Arabic parsing—including the experiments in the previous section—have assumed gold segmentation. But gold segmentation is not available in application settings, so a segmenter and parser are arranged in a pipeline. Segmentation errors cascade into the parsing phase, placing an artificial limit on parsing performance.

Lattice parsing (Chappelier et al., 1999) is an alternative to a pipeline that prevents cascading errors by placing all segmentation options into the parse chart. Recently, lattices have been used successfully in the parsing of Hebrew (Tsarfaty, 2006; Cohen and Smith, 2007), a Semitic language with similar properties to Arabic. We extend the Stanford parser to accept pre-generated lattices, where each word is represented as a finite state automaton. To combat the proliferation of parsing edges, we prune the lattices according to a hand-constructed lexicon of 31 clitics listed in the ATB annotation guidelines (Maamouri et al., 2009a). Formally, for a lexicon L and segments $I \in L$, $O \notin L$, each word automaton accepts the language $I^*(O+I)I^*$. Aside from adding a simple rule to correct *alif* deletion caused by the preposition ل, no other language-specific processing is performed.

Our evaluation includes both weighted and unweighted lattices. We weight edges using a unigram language model estimated with Good-Turing smoothing. Despite their simplicity, unigram weights have been shown as an effective feature in segmentation models (Dyer, 2009).[13] The joint parser/segmenter is compared to a pipeline that uses MADA (v3.0), a state-of-the-art Arabic segmenter, configured to replicate ATB segmentation (Habash and Rambow, 2005). MADA uses an ensemble of SVMs to first re-rank the output of a deterministic morphological analyzer. For each

[13]Of course, this weighting makes the PCFG an improper distribution. However, in practice, unknown word models also make the distribution improper.

Label	# gold	F1
ADJP	**1216**	**59.45**
SBAR	**2918**	**69.81**
FRAG	254	72.87
VP	5507	78.83
S	6579	78.91
PP	7516	80.93
NP	34025	84.95
ADVP	1093	90.64
WHNP	787	96.00

(a) Major phrasal categories

Tag	# gold	%	Tag	# gold	%
VBG	**182**	**48.84**	**JJR**	**134**	**92.83**
VN	**163**	**60.37**	DTNNS	1069	94.29
VBN	352	72.42	**DTJJ**	**3361**	**95.07**
DTNNP	932	83.48	NNP	4152	95.09
JJ	**1516**	**86.09**	NN	10336	95.23
ADJ_NUM	**277**	**88.93**	DTNN	6736	95.78
VBP	2139	89.94	NOUN_QUANT	352	98.16
RP	818	91.23	PRP	1366	98.24
NNS	907	91.75	CC	4076	98.92
DTJJR	**78**	**92.41**	IN	8676	99.07
VBD	2580	92.42	DT	525	99.81

(b) Major POS categories

Parent	Head	Modifer	Dir	# gold	F1
NP	NP	TAG	R	946	0.54
S	**S**	**S**	**R**	**708**	**0.57**
NP	**NP**	**ADJP**	**R**	**803**	**0.64**
NP	**NP**	**NP**	**R**	**2907**	**0.66**
NP	NP	SBAR	R	1035	0.67
NP	**NP**	**PP**	**R**	**2713**	**0.67**
VP	TAG	PP	R	3230	0.80
NP	NP	TAG	L	805	0.85
VP	TAG	SBAR	R	772	0.86
S	VP	NP	L	961	0.87

(c) Ten lowest scoring (Collins, 2003)-style dependencies occurring more than 700 times

Table 8: Per category performance of the Berkeley parser on sentence lengths ≤ 70 (dev set, gold segmentation). **(a)** Of the high frequency phrasal categories, ADJP and SBAR are the hardest to parse. We showed in §2 that lexical ambiguity explains the underperformance of these categories. **(b)** POS tagging accuracy is lowest for *maSdar* verbal nouns (VBG,VN) and adjectives (e.g., JJ). Richer tag sets have been suggested for modeling morphologically complex distinctions (Diab, 2007), but we find that linguistically rich tag sets do not help parsing. **(c)** Coordination ambiguity is shown in dependency scores by e.g., ⟨S S S R⟩ and ⟨NP NP NP R⟩. ⟨NP NP PP R⟩ and ⟨NP NP ADJP R⟩ are both *iDafa* attachment.

input token, the segmentation is then performed deterministically given the 1-best analysis.

Since guess and gold trees may now have different yields, the question of evaluation is complex. Cohen and Smith (2007) chose a metric like SParseval (Roark et al., 2006) that first aligns the trees and then penalizes segmentation errors with an edit-distance metric. But we follow the more direct adaptation of Evalb suggested by Tsarfaty (2006), who viewed exact segmentation as the ultimate goal. Therefore, we only score guess/gold pairs with identical *character* yields, a condition that allows us to measure parsing, tagging, and segmentation accuracy by ignoring whitespace.

Table 9 shows that MADA produces a high quality segmentation, and that the effect of cascading segmentation errors on parsing is only 1.92% F1. However, MADA is language-specific and relies on manually constructed dictionaries. Conversely, the lattice parser requires no linguistic resources and produces segmentations of comparable quality. Nonetheless, parse quality is much lower in the joint model because a lattice is effectively a long sentence. A cell in the bottom row of the parse chart is required for each potential whitespace boundary. As we have said, parse quality decreases with sentence length. Finally, we note that simple weighting gives nearly a 2% F1 improvement, whereas Goldberg and Tsarfaty (2008) found that unweighted lattices were more effective for Hebrew.

	LP	LR	F1	Seg F1	Tag F1	Coverage
STANFORD (Gold)	81.64	80.55	81.09	100.0	95.81	100.0%
MADA	—	—	—	97.67	—	96.42%
MADA+STANFORD	**79.44**	**78.90**	**79.17**	**97.67**	**94.27**	**96.42%**
STANFORDJOINT	76.13	72.61	74.33	94.12	90.13	94.73%
STANFORDJOINT+UNI	77.09	74.97	76.01	96.26	92.23	95.87%

Table 9: Dev set results for sentences of length ≤ 70. Coverage indicates the fraction of hypotheses in which the character yield exactly matched the reference. Each model was able to produce hypotheses for all input sentences. In these experiments, the input lacks segmentation markers, hence the slightly different dev set baseline than in Table 6.

7 Conclusion

By establishing significantly higher parsing baselines, we have shown that Arabic parsing performance is not as poor as previously thought, but remains much lower than English. We have described grammar state splits that significantly improve parsing performance, catalogued parsing errors, and quantified the effect of segmentation errors. With a human evaluation we also showed that ATB inter-annotator agreement remains low relative to the WSJ corpus. Our results suggest that current parsing models would benefit from better annotation consistency and enriched annotation in certain syntactic configurations.

Acknowledgments We thank Steven Bethard, Evan Rosen, and Karen Shiells for material contributions to this work. We are also grateful to Markus Dickinson, Ali Farghaly, Nizar Habash, Seth Kulick, David McCloskey, Claude Reichard, Ryan Roth, and Reut Tsarfaty for constructive discussions. The first author is supported by a National Defense Science and Engineering Graduate (NDSEG) fellowship. This paper is based on work supported in part by DARPA through IBM. The content does not necessarily reflect the views of the U.S. Government, and no official endorsement should be inferred.

References

Al-Batal, M. 1990. Connectives as cohesive elements in a modern expository Arabic text. In Eid, Mushira and John McCarthy, editors, *Perspectives on Arabic Linguistics II*. John Benjamins.

Arun, A and F Keller. 2005. Lexicalization in crosslinguistic probabilistic parsing: The case of French. In *ACL*.

Bikel, D M. 2004. Intricacies of Collins' parsing model. *Computational Linguistics*, 30:479–511.

Chappelier, J-C, M Rajman, R Arages, and A Rozenknop. 1999. Lattice parsing for speech recognition. In *TALN*.

Chiang, D, M Diab, N Habash, O Rambow, and S Shareef. 2006. Parsing Arabic dialects. In *EACL*.

Clegg, A and A Shepherd. 2005. Evaluating and integrating treebank parsers on a biomedical corpus. In *ACL Workshop on Software*.

Cohen, S and N A Smith. 2007. Joint morphological and syntactic disambiguation. In *EMNLP*.

Collins, M, J Hajic, L Ramshaw, and C Tillmann. 1999. A statistical parser for Czech. In *ACL*.

Collins, M. 2003. Head-Driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

Cowan, B and M Collins. 2005. Morphology and reranking for the statistical parsing of Spanish. In *NAACL*.

Diab, M. 2007. Towards an optimal POS tag set for Modern Standard Arabic processing. In *RANLP*.

Dickinson, M. 2005. *Error Detection and Correction in Annotated Corpora*. Ph.D. thesis, The Ohio State University.

Dubey, A and F Keller. 2003. Probabilistic parsing for German using sister-head dependencies. In *ACL*.

Dyer, C. 2009. Using a maximum entropy model to build segmentation lattices for MT. In *NAACL*.

Fassi Fehri, A. 1993. *Issues in the structure of Arabic clauses and words*. Kluwer Academic Publishers.

Gabbard, R and S Kulick. 2008. Construct state modification in the Arabic treebank. In *ACL*.

Goldberg, Y and R Tsarfaty. 2008. A single generative model for joint morphological segmentation and syntactic parsing. In *ACL*.

Green, S, C Sathi, and C D Manning. 2009. NP subject detection in verb-initial Arabic clauses. In *Proc. of the Third Workshop on Computational Approaches to Arabic Script-based Languages (CAASL3)*.

Habash, N and O Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *ACL*.

Habash, N and O Rambow. 2007. Arabic diacritization through full morphological tagging. In *NAACL*.

Habash, N and R Roth. 2009. CATiB: The Columbia Arabic Treebank. In *ACL, Short Papers*.

Habash, N and F Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *NAACL*.

Habash, N, B Dorr, and C Monz. 2006. Challenges in building an Arabic-English GHMT system with SMT components. In *EAMT*.

Hajič, J and P Zemánek. 2004. Prague Arabic dependency treebank: Development in data and tools. In *NEMLAR*.

Huang, Z and M Harper. 2009. Self-training PCFG grammars with latent annotations across languages. In *EMNLP*.

Klein, D and C D Manning. 2002. Fast exact inference with a factored model for natural language parsing. In *NIPS*.

Klein, D and C D Manning. 2003. Accurate unlexicalized parsing. In *ACL*.

Kulick, S, R Gabbard, and M Marcus. 2006. Parsing the Arabic Treebank: Analysis and improvements. In *TLT*.

Kübler, S. 2005. How do treebank annotation schemes influence parsing results? Or how not to compare apples and oranges. In *RANLP*.

Levy, R and C D Manning. 2003. Is it harder to parse Chinese, or the Chinese treebank? In *ACL*.

Maamouri, M and A Bies. 2004. Developing an Arabic Treebank: Methods, guidelines, procedures, and tools. In *Proc. of the Workshop on Computational Approaches to Arabic Script-based Languages (CAASL1)*.

Maamouri, M, A Bies, T Buckwalter, and W Mekki. 2004. The Penn Arabic Treebank: Building a large-scale annotated Arabic corpus. In *NEMLAR*.

Maamouri, M, A Bies, and S Kulick. 2008. Enhancing the Arabic Treebank: A collaborative effort toward new annotation guidelines. In *LREC*.

Maamouri, M, A Bies, S Krouna, F Gaddeche, and B Bouziri. 2009a. Penn Arabic Treebank guidelines v4.92. Technical report, Linguistic Data Consortium, University of Pennsylvania, August 5.

Maamouri, M, A Bies, and S Kulick. 2009b. Creating a methodology for large-scale correction of treebank annotation: The case of the Arabic Treebank. In *MEDAR*.

Marcus, M, M A Marcinkiewicz, and B Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330.

Petrov, S, L Barrett, R Thibaux, and D Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *ACL*.

Petrov, S. 2009. *Coarse-to-Fine Natural Language Processing*. Ph.D. thesis, University of California-Berkeley.

Rehbein, I and J van Genabith. 2007. Treebank annotation schemes and parser evaluation for German. In *EMNLP-CoNLL*.

Roark, B, M Harper, E Charniak, B Dorr, M Johnson, J G Kahne, Y Liuf, Mari Ostendorf, J Hale, A Krasnyanskaya, M Lease, I Shafran, M Snover, R Stewart, and L Yung. 2006. SParseval: Evaluation metrics for parsing speech. In *LREC*.

Ryding, K. 2005. *A Reference Grammar of Modern Standard Arabic*. Cambridge University Press.

Sampson, G and A Babarczy. 2003. A test of the leaf-ancestor metric for parse accuracy. *Natural Language Engineering*, 9:365–380.

Toutanova, K, D Klein, C D Manning, and Y Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL*.

Tsarfaty, R and K Sima'an. 2008. Relational-realizational parsing. In *COLING*.

Tsarfaty, R. 2006. Integrated morphological and syntactic disambiguation for Modern Hebrew. In *ACL*.

Zitouni, I, J S Sorensen, and R Sarikaya. 2006. Maximum entropy based restoration of Arabic diacritics. In *ACL*.

Paraphrase Alignment for Synonym Evidence Discovery

Gintarė Grigonytė
Saarland University
g.grigonyte@hmf.vdu.lt

João Cordeiro, Gaël Dias, Rumen Moraliyski
HULTIG
University of Beira Interior
{jpaulo,ddg,rumen}@di.ubi.pt

Pavel Brazdil
LIAAD/FEP
University of Porto
pbrazdil@liaad.up.pt

Abstract

We describe a new unsupervised approach for synonymy discovery by aligning paraphrases in monolingual domain corpora. For that purpose, we identify phrasal terms that convey most of the concepts within domains and adapt a methodology for the automatic extraction and alignment of paraphrases to identify *paraphrase casts* from which valid synonyms are discovered. Results performed on two different domain corpora show that general synonyms as well as synonymic expressions can be identified with a 67.27% precision.

1 Introduction

Synonymy is a specific type of a semantic relationship. According to (Sowa and Siekmann, 1994), a synonym is a word (or concept) that means the same or nearly the same as another word (or concept). It has been observed that words are similar if their contexts are similar (Freitag et al., 2005) and so synonymy detection has received a lot of attention during the last decades. However, words used in the same context are not necessarily synonyms and can embody different semantic relationships such as hyponyms, meronyms or co-hyponyms (Heylen et al., 2008). In this paper, we introduce a new unsupervised methodology for synonym detection by extracting and aligning paraphrases on normalized domain corpora[1]. In particular, we study a specific structure within aligned paraphrases, *paraphrase casts*, from which valid synonyms are discovered. In fact, we propose a new approach based on the idea that synonyms are substitutable words within a given domain corpus. Results performed on two different domain corpora, the Corpus of Computer Security (COCS) and the Corpus of Cancer Research (COCR), show that general synonyms as well as synonymic expressions can be identified with a 67.27% precision performance.

[1]By normalized, we intend that phrasal terms have been previously identified.

2 Related Work

Automatic synonymy detection has been tackled in a variety of ways which we explain as follows.

2.1 Pattern-based Approaches

This approach to information extraction is based on a technique called *selective concept extraction* as defined by (Riloff, 1993). Selective concept extraction is a form of text skimming that selectively processes relevant text while effectively ignoring surrounding text that is thought to be irrelevant to the domain. The pioneer of pattern-based approaches (Hearst, 1992) has introduced lexico-syntactic patterns to automatically acquire given word semantic relationships. Specific patterns like "*X and other Y*" or "*X such as Y*" were used for hypernym-hyponym detection. Later, the idea was extended and adapted for synonymy by other researchers such as (Roark and Charniak, 1998), (Caraballo, 1999) and (Maynard and Peters, 2009). In general, manual pattern definition is time consuming and requires linguistic skills. Usually, systems based on lexico-syntactic patterns perform with very high precision, but low recall due to the fact that these patterns are rare. However, recent work by (Ohshima and Tanaka,

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 403–411,
Beijing, August 2010

2009) on Web data reported high recall figures. To avoid manual encoding of patterns, many supervised approaches have been proposed as summarized in (Stevenson and Greenwood, 2006).

2.2 Distributional Similarity

Distributional similarity for capturing semantic relatedness is relying on the hypothesis that semantically similar words share similar contexts. These methods vary in the level of supervision from unsupervised to semi-supervised or to supervised. The first type of methods includes the work of (Hindle, 1990), (Lin, 1998) and (Heylen et al., 2008) who used unsupervised methods for detecting word similarities based on shallow-parsed corpora. Others have proposed unsupervised methodologies to solve TOEFL-like tests, instead of discovering synonyms (Turney, 2001), (Terra and Clarke, 2003) and (Freitag et al., 2005). Other researchers, such as (Girju et al., 2004), (Muller et al., 2006), (Wu and Zhou, 2003) and (Wei et al., 2009), have used language or knowledge resources to enhance the representation of the vector space model. Unlike the pattern-based approach, the distributional similarity-based approach shows low precision compared to high recall.

One obvious way to verify all the possible connections between words of the vocabulary employs an exhaustive search. However, comparison based on word usage can only highlight those terms that are highly similar in meaning. This method of representation is usually unable to distinguish between middle strength and weak semantic relations, or detect the relationship between hapax-legomena.

2.3 Hybrid Approaches

More recently, approaches combining patterns and distributional similarity appeared to bring the best of the two methodologies. (Hagiwara et al., 2009) describe experiments that involve training various synonym classifiers. (Giovannetti et al., 2008) use syntactically parsed text and manually composed patterns together with distributional similarity for detecting semantically related words. Finally, (Turney, 2008) proposes a supervised machine learning approach for discovering synonyms, antonyms, analogies and associations. For that purpose, feature vectors are based on frequencies of patterns and classified by a SVM.

2.4 Our Approach

(Van der Plas and Tiedemann, 2006) state that "*People use multiple ways to express the same idea. These alternative ways of conveying the same information in different ways are referred to by the term* paraphrase *and in the case of single words or phrasal terms sharing the same meaning, we speak of synonyms*". Based on this, we propose that in order to discover pairs of semantically related words (in the best case synonyms) that may be used in figurative or rare sense, and as consequence impossible to be identified by the distributional similarity approach, we need to have them highlighted by their **local specific** environment. Here we differ from the pattern-based approach that use **local general** environment. We propose to align paraphrases from domain corpora and discover words that are possibly substitutable for one another in a given context (*paraphrase casts*), and as such are synonyms or near-synonyms. Comparatively to existing approaches, we propose an unsupervised and language-independent methodology which does not depend on linguistic processing[2], nor manual definition of patterns or training sets and leads to higher precision when compared to distributional similarity-based approaches.

3 Normalization of the Corpora

The main goal of our research is to build knowledge resources in different domains that can effectively be used in different NLP applications. As such, precision takes an important part in the overall process of our methodology. For that purpose, we first identify the phrasal terms (or multi-word units) present in the corpora. Indeed, it has been shown in many works that phrasal terms convey most of the specific contents of a given domain. Our approach to term extraction is based on linguistic pattern matching and Inverse Document Frequency (IDF) measurements for term

[2] We will see in the next section that we will use linguistic resources to normalize our corpora, but the methodology can be applied to any raw text.

quality assurance as explained in (Avizienis et al., 2009). For that purpose, we present a domain independent hybrid term extraction framework that includes the following steps. First, the text is morphologically annotated with the MPRO system (Maas et al., 2009). Then grammar rules for morphological disambiguation, syntactic parsing and noun phrase detection are applied based on finite-state automata technology, KURD (Carl and Schmidt-Wigger, 1998). Following this, a variant and non-basic term form detection is applied, as well as stop words removal. Then, combining rich morphological and shallow syntactical analysis with pattern matching techniques allows us to extract a wide span of candidate terms which are finally filtered with the well-known IDF measure.

4 Paraphrase Identification

A few unsupervised metrics have been applied to automatic paraphrase identification and extraction (Barzilay and McKeown, 2001) and (Dolan et al., 2004). However, these unsupervised methodologies show a major drawback by extracting quasi-exact or even exact match pairs of sentences as they rely on classical string similarity measures. Such pairs are useless for our research purpose. More recently, (Cordeiro et al., 2007a) proposed the *sumo* metric specially designed for asymmetrical entailed pair identification in corpora which obtained better performance than previously established metrics, even in corpora with exclusively symmetrical entailed paraphrases as in the Microsoft Paraphrase Research Corpus (Dolan et al., 2004). This function states that for a given sentence pair $\langle S_a, S_b \rangle$, having m and n words in each sentence and λ lexical exclusive links (word overlaps, see figure 1) between them, its lexical connection strength is computed as defined in Equations 1 and 2.

$$Sumo(S_a, S_b) = \begin{cases} S(m,n,\lambda) & if \;\; S(m,n,\lambda) < 1 \\ 0 & if \;\; \lambda = 0 \\ e^{-kS(m,n,\lambda)} & otherwise \end{cases} \quad (1)$$

where

$$S(m,n,\lambda) = \alpha \log_2(\tfrac{m}{\lambda}) + \beta \log_2(\tfrac{n}{\lambda}) \\ \alpha, \beta \in [0,1], \alpha + \beta = 1 \quad (2)$$

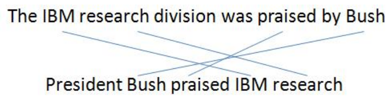


Figure 1: 4 exclusive links between S_a and S_b.

To obtain a paraphrase corpus, we compute all sentence pairs similarities $Sumo(S_a, S_b)$ and select only those pairs exceeding a given threshold, in our case $threshold = 0.85$, which is quite restrictive, ensuring the selection of pairs strongly connected[3].

However, to take into account the normalization of the corpus, little adjustments had to be integrated in the methodology proposed in (Cordeiro et al., 2007a). Indeed, the original $Sumo(.,.)$ function was under-weighting links that occurred between phrasal terms such as "*molecular laboratory*" or "*renal cancer*". So, instead of counting the number of lexical links among sentences, each link weights differently according to the word length in the connection, hence connections of longer words will result in a larger value. For example, in figure 1, instead of having $\lambda = 4$, we count $\lambda = 3 + 8 + 7 + 4 = 22$. This adjustment is important as multi-word units are treated as longer words in the corpus. This modification has also, as a side effect, under-evaluation of functional words which usually follow the Zipf's Law and give more importance to meaningful words in the paraphrase extraction process.

5 Paraphrase Alignment

In order to usefully explore the evidence synonymy from paraphrases, sentence alignment techniques must be applied to paraphrases in order to identify *paraphrase casts*, i.e., substitutable word pairs as shown in figure 2. As we can see, the paraphrase cast includes three parts: the left segment (context), a middle segment and the right segment (context). In our figure the left and right segments (contexts) are identical.

In the context of DNA sequence alignment, two main algorithms have been proposed: (1) the Needleman-Wunsch algorithm (Needleman and

[3]Further details about the sumo metric are available in (Cordeiro et al., 2007a).

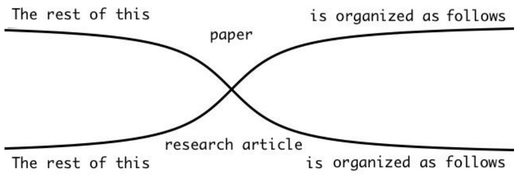


Figure 2: A paraphrase cast.

Wunsch, 1970) based on dynamic programming which outputs a unique global alignment and (2) the Smith-Waterman (SW) algorithm (Smith and Waterman, 1981) which is an adaptation of the previous algorithm and outputs local alignments. In the context of NLP, (Cordeiro et al., 2007a) proposed a combination of both algorithms depending on the structure of paraphrase. However, since any local alignment is a candidate for *paraphrase casts*, the SW algorithm revealed itself more appropriate and was always chosen. The SW alignment algorithm uses dynamic programming to compute the optimal local alignments between two sequences[4]. This process requires first the definition of an alignment matrix (function), which governs the likelihood of alignment of two symbols. Thus we first build a matrix H such that $H(i, 0) = 0$ and $H(0, j) = 0$, for $0 \leq i \leq m$, and $0 \leq j \leq n$, where m and n are the number of words in the paraphrase sentences. The rest of the H elements are recursively calculated as in Equation 3 where $gs(.,.)$ is the gap-scoring function and S_{a_i} (resp. S_{b_j}) represents the i^{th} (resp. j^{th}) word of sentence S_a (resp. S_b).

$$H(i,j) = max \begin{cases} 0 & \\ H(i-1, j-1) + gs(S_{a_i}, S_{b_j}) & \text{MMisatch} \\ H(i-1, j) + gs(S_{a_i}, \_) & \text{Deletion} \\ H(i, j-1) + gs(\_, S_{b_j}) & \text{Insertion} \end{cases} \tag{3}$$

Obviously, this algorithm is based on an alignment function which exploits the alignment likelihood between two alphabet symbols. For DNA sequence alignments, this function is defined as a mutation matrix, scoring gene mutation and gap alignments. In our case, we define the gap-scoring function $gs(.,.)$ in Equations 4 and 5 which prioritize the alignment of specific domain key terms i.e., single match, or key expressions i.e., phrasal match, (reward 50), as well as lexically similar[5] words such as "*programme*" and "*programming*" for example. In particular, for these similar words an adaptation of the well known *Edit Distance* is used, the $c(.,.)$ function (5), which is explained in more detail in (Cordeiro et al., 2007b).

$$gs(S_{a_i}, S_{b_j}) = \begin{cases} -1 & \text{if } (S_{a_i} = -) \text{ and } (S_{b_j} \neq -) \\ -1 & \text{if } (S_{b_j} = -) \text{ and } (S_{a_i} \neq -) \\ 10 & \text{Single Match} \\ 50 & \text{Phrasal Match} \\ c(S_{a_i}, S_{b_j}) & \text{Mismatch} \end{cases} \tag{4}$$

where

$$c(S_{a_i}, S_{b_j}) = -\frac{edist(S_{a_i}, S_{b_j})}{\epsilon + maxseq(S_{a_i}, S_{b_j})} \tag{5}$$

To obtain local alignments, the SW algorithm is applied, using the alignment function defined with $H(.,.)$ in 3. The alignment of the paraphrase in figure 2 would give the result in figure 3.

```
The rest of this paper ---------------- is organized as follows
The rest of the  ----- research article is organized as follows
```

Figure 3: An alignment.

6 Paraphrase Casts

In order to discover synonyms, we are looking for special patterns from the aligned paraphrase sentences, which naturally give us more evidence for the existence of equivalent terms or expressions. Due to the topological aspect of such patterns, we decided to name them *paraphrase casts*, or just *casts* as shown in figure 2. As we have mentioned earlier, the paraphrase cast includes three parts: the left segment (*contextL*), a middle segment and the right segment (*contextR*). In the following example the left and right segments (contexts) are identical, but the middle segment includes **different** misaligned sequences of words, represented by *wordSa* and *wordSb*.

```
contextL wordSa ----- contextR
contextL ----- wordSb contextR
```

[4]In our case, the two sequences are the two sentences of a paraphrase

[5]This is why we have in equation 3 the label "Mismatch", where "mismatch" means different yet lexically near words.

We can attribute different levels of confidence to different paraphrase casts. Indeed, the larger the contexts and the smaller the misaligned sequences are, the more likely it is for single or phrasal terms to be synonyms or near-synonyms. Note that in the cast shown in figure 3, each context has a significant size, with four words on each side, and the misaligned segments are in fact equivalent expressions i.e. "*paper*" is a synonym of "*research article*". In the analyzed domain these expressions are equivalent and interchangeable and appear to be interchangeable in other domains. For the purpose of this paper, we only take into account the casts where the misaligned sequences of words contain only one word or one multi-word unit in each sentence. That is, we have a one-to-one match. However, no constraints are imposed on the contexts[6]. So, all casts are computed and analyzed for synonym discovery and results are provided in the next section.

7 Experiments

To evaluate our methodology we have used two different corpora, both from scientific domains built from abstracts of publications (see Table 1). The corpus of computer security (COCS) is a collection of 4854 abstracts on computer security extracted from the IEEE (http://ieee.rkbexplorer.com/) repository[7]. The corpus of cancer research (COCR) contains 3334 domain specific abstracts of scientific publications extracted from the PubMed[8] on three types of cancer: (1) the mammary carcinoma register (COCR1) consisting of 1500 abstracts, (2) the nephroblastoma register (COCR2) consisting of 1500 abstracts, and (3) the rhabdoid tumor register (COCR3) consisting of 334 abstracts. From the paraphrase casts, we were able to automatically extract, without further processing, single synonymous word pairs, as well as synonymic multi-word units, as can be seen in Table 2. For that purpose we have used specific paraphrase casts, whose aim was to privilege precision to recall. In particular, we have removed all casts which contained numbers or special characters i.e. casts with filter in Table 1. However, no constraints were imposed on the frequency of the casts. Indeed, all casts were included even if their overall frequency was just one. Although most of the word relationships were concerned with synonymy, the other ones were not just errors, but lexically related words, namely examples of antonymy, hyperonymy/hyponymy and associations as shown in Table 3. In the evaluation, we have focused on the precision of the method. The evaluation of the extracted pairs was performed manually by two domain experts. In fact, four

Corpus	COCS	COCR1	COCR2	COCR3
Tokens	412.265	336.745	227.477	46.215
Sentences	18.974	15.195	10.575	2.321
Aligned Pairs	589	994	511	125
Casts without filter	320	10.217	2.520	48
Casts with filter	202	361	292	16

Table 1: Corpora

Synonym (COCS)	**Complementary**
frequency tuning	frequency control
attack consequences	attack impact
error-free operation	error free operation
pseudo code	pseudo algorithm
tolerance	resilience
package loss	message loss
adjustable algorithm	context-aware algorithm
helpful comment	valuable comment
Synonym (COCR)	**Complementary**
childhood renal tumor	childhood kidney tumor
hypertrophy	growth
doxorubicin	vincristine
carcinomas of the kidney	sarcoma of the kidney
metastasis	neoplasm
renal tumor	renal malignancy
neoplastic thrombus	tumor thrombus
vincristine	adriamycin

Table 2: Synonyms for COCS

Antonym	**Complementary**
positive sentinel nodes	negative sentinel nodes
higher bits	lower bits
older version	newer version
Hypernym	**Hyponym**
Multi-Tasking Virtual Machine	Java Virtual Machine
therapy	chemotherapy
hormone breast cancer	estrogen breast cancer
Association	**Complementary**
performance	reliability
statistical difference	significant difference
relationship	correlation

Table 3: Other Word Semantic Relationships.

[6]This issue will be discussed in the next section.

[7]An example of an abstract can be viewed at: http://ieee.rkbexplorer.com/description/publication-00534618

[8]http://www.ncbi.nlm.nih.gov/pubmed

different evaluations were carried out depending on whether the adapted $S(.,.)$ measure was used (or not) and whether the normalization of the corpora was used (or not). The best results were obtained in all cases for the adapted $S(.,.)$ measure with the multi-word units. Table 4 shows also the worst results for the COCS as a baseline (COCS (1)), i.e. non-adapted $S(.,.)$ and non-normalized corpus. For the rest of the experiments we always present the results with the adapted $S(.,.)$ measure and normalized corpus.

Corpus	**COCS (1)**	**COCS (2)**	
Precision	54.62%	71.29%	
Extracted Synonyms	130	144	
Errors	108	58	
Corpus	**COCR1**	**COCR2**	**COCR3**
Precision	69.80%	61.30%	75.00%
Extracted Synonyms	252	178	12
Errors	109	111	4

Table 4: Overall Results

7.1 Discussion

Many results have been published in the literature, especially tackling the TOEFL synonym detection problem which aims at identifying the correct synonym among a small set of alternatives (usually four). For that purpose, the best precision rate has been reached by (Turney et al., 2003) with 97.50% who have exploited many resources, both statistical and linguistic. However, our methodology tackles **a different problem**. Indeed, our goal is to automatically extract synonyms from texts. The published works toward this direction have not reached so good results. One of the latest studies was conducted by (Heylen et al., 2008) who used distributional similarity measures to extract synonyms from shallow parsed corpora with the help of unsupervised methods. They report that "*the dependency-based model finds a tightly related neighbor for 50% of the target words and a true synonym for 14%*". So, it means that by comparing all words in a corpus with all other words, one can expect to find a correct semantic relationship in 50% of the cases and a correct synonym in just 14%. In that perspective, our approach reaches higher results. Moreover, (Heylen et al., 2008) use a frequency cut-off which only selects features that occur at least five times together with the target word. In our case, no frequency threshold is imposed to enable extraction of synonyms with low frequency, such as *hapax legomena*. This situation is illustrated in figure 4. We note that most of the candidate pairs appear only once in the corpus.

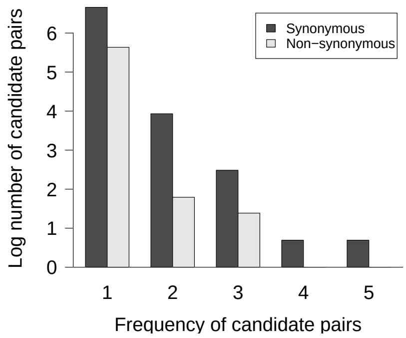


Figure 4: Synonyms Frequency Distribution.

In order to assess the quality of our results, we calculated the similarity between all extracted pairs of synonyms following the distributional analysis paradigm as in (Moraliyski and Dias, 2007) who build context[9] feature vectors for noun synonyms. In particular, we used the cosine similarity measure and the Loglike association measure (Dunning, 1993) as the weighting scheme of the context features. The distribution of the similarity measure for all noun synonyms (62 pairs) is shown in figure 5.

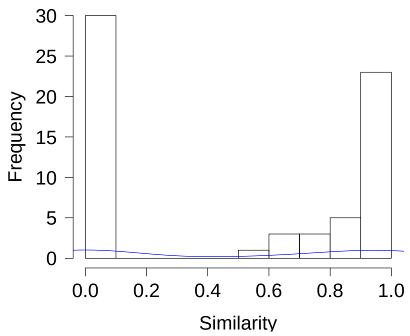


Figure 5: Synonym Pairs Similarity Distribution.

The results clearly show that all extracted synonyms are highly correlated in terms of context.

[9] In this case, the contexts are the surrounding nouns, verbs and adjectives in the closest chunks of a shallow parsed corpus.

Nearly half of the cases have similarities higher than 0.5. It is important to notice that a specific corpus[10] was built to calculate as sharply as possible the similarity measures as it is done in (Moraliyski and Dias, 2007). Indeed, based on the COCS and the COCR most statistics were insignificant leading to zero-valued features. This situation is well-known as it is one of the major drawbacks of the distributional analysis approach which needs huge quantities of texts to make secure decisions. So we note that applying the distributional analysis approach to such small corpora would have led to rather poor results. Even with an adapted corpus, figure 5 (left-most bar) shows that there are no sufficient statistics for 30 pairs of synonyms. Although the quality of the extracted pairs of synonyms is high, the precision remains relatively low with 67.27% precision on average. As we pointed out in the previous section, we did not make any restrictions to the left and right contexts of the casts. However, the longer these contexts are, compared to the misaligned sequence of words, the higher the chance is that we find a correct synonym. Table 5 shows the average lengths of both cast contexts for synonyms and erroneous pairings, in terms of words (WCL) and characters (CCL). We also provide the ratio (R) between the character lengths of the middle segment (i.e. misaligned character sequences) and the character lengths of the cast contexts (i.e. right and left sizes of equally aligned character sequences). It is clear that a more thorough study of the effects of the left and right contexts should be carried out to improve precision of our approach, although this may be to the detriment of recall. Based on the results of the ratio R[11], we note that the larger the cast context is compared to the cast content, the more likely it is that the misaligned words are synonyms.

Type	WCL	CCL	R
Synonyms	2.70	11.67	0.70
Errors	2.45	8.05	0.55

Table 5: Average Casts Contexts Lengths

[10] This corpus contains 125.888.439 words.

[11] These results are statistically relevant with $p - value < 0.001$ using the Wilcoxon Rank-Sum Test.

8 Conclusions

In this paper we have introduced a new unsupervised methodology for synonym detection that involves extracting and aligning paraphrases on normalized domain corpora. In particular, we have studied a specific structure within aligned paraphrases, *paraphrase casts*, from which valid synonyms were discovered. The overall precision was 71.29% for the computer security domain and 66.06% for the cancer research domain. This approach proved to be promising for extracting synonymous words and synonymic multiword units. Its strength is the ability to effectively work with small domain corpora, without supervised training, nor definition of specific language-dependent patterns. Moreover, it is capable to extract synonymous pairs with figurative or rare senses which would be impossible to identify using the distributional similarity approach. Finally, our approach is completely language-independent as it can be applied to any raw text, not obligatorily normalized corpora, although the results for domain terminology may be improved by the identification of phrasal terms.

However, further improvements of the method should be considered. A measure of quality of the *paraphrase casts* is necessary to provide a measure of confidence to the kind of extracted word semantic relationship. Indeed, the larger the contexts and the smaller the misaligned sequences are, the more likely it is for single or phrasal terms to be synonyms or near-synonyms. Further work should also be carried out to differentiate the acquired types of semantically related pairs. As it is shown in Table 3, some of the extracted pairs were not synonymic, but lexically related words such as antonyms, hypernyms/hyponyms and associations. A natural follow-up solution for discriminating between semantic types of extracted pairs could involve context-based classification of acquired *casts* pairs. In particular, (Turney, 2008) tackled the problem of classifying different lexical information such as synonymy, antonymy, hypernymy and association by using context words. In order to propose a completely unsupervised methodology, we could also follow the idea of (Dias et al., 2010) to automatically construct small

TOEFL-like tests based on sets of *casts* which could be handled with the help of different distributional similarities.

Acknowledgments

We thank anonymous reviewers whose comments helped to improve this paper. We also thank IFOMIS institute (Saarbrucken) and the ReSIST project for allowing us to use the COCR and COCS corpora.

References

Avizienis, A., Grigonyte, G., Haller, J., von Henke, F., Liebig, T., and Noppens, O. 2009. *Organizing Knowledge as an Ontology of the Domain of Resilient Computing by Means of NLP - An Experience Report.* In Proc. of the 22th Int. FLAIRS Conf. AAAI Press, pp. 474-479.

Barzilay, R. and McKeown, K. R. 2001. *Extracting Paraphrases from a Parallel Corpus.* In Proc. of the 39th meeting of ACL, pp. 50-57.

Caraballo, S. A. 1999. *Automatic Construction of a Hypernym-labeled Noun Hierarchy from Text.* In Proc. of 37th meeting of ACL 1999, pp 120-126.

Carl, M., and Schmidt-Wigger, A. 1998. *Shallow Post Morphological Processing with KURD.* In Proc. of the Conf. on New Methods in Language Processing.

Cordeiro, J.P., Dias, G. and Brazdil, P. 2007. *Learning Paraphrases from WNS Corpora.* In Proc. of the 20th Int. FLAIRS Conf. AAAI Press, pp. 193-198.

Cordeiro, J.P., Dias, G. and Cleuziou, G. 2007. *Biology Based Alignments of Paraphrases for Sentence Compression.* In Proc. of the 20th meeting of ACL, workshop PASCAL, pp. 177-184.

Dias, G., Moraliyski, R., Cordeiro, J.P., Doucet, A. and Ahonen-Myka, H. 2010. *Automatic Discovery of Word Semantic Relations using Paraphrase Alignment and Distributional Lexical Semantics Analysis.* In Journal of Natural Language Engineering, Cambridge University Press. ISSN 1351-3249, pp. 1-26.

Dolan, B., Quirk, C. and Brockett, C. 2004. *Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources.* In Proc. of the 20th int. Conf. on Computational Linguistics.

Dunning T. D 1993. *Accurate Methods for the Statistics of Surprise and Coincidence.* In Computational Linguistics, 19(1), pp. 61-74.

Freitag, D., Blume, M., Byrnes, J., Chow, E., Kapadia, S., Rohwer, R. and Wang, Z. 2005. *New Experiments in Distributional Representations of Synonymy.* In Proc. of 9th conf. on Computational Natural Language Learning, pp. 25-32.

Giovannetti, E., Marchi, S. and Montemagni, S. 2008. *Combining Statistical Techniques and Lexico-Syntactic Patterns for Semantic Relations Extraction from Text.* In Proc. of the 5th Workshop on Semantic Web Applications and Perspectives.

Girju, R., Giuglea, A. M., Olteanu, M., Fortu, O., Bolohan, O. and Moldovan, D. 2004. *Support Vector Machines Applied to the Classification of Semantic Relations in Nominalized Noun Phrases.* In Proc. of the HLT-NAACL Workshop on Computational Lexical Semantics, pp. 68-75.

Hagiwara, M. O. Y. and Katsuhiko, T. 2009. *Supervised Synonym Acquisition using Distributional Features and Syntactic Patterns.* In Information and Media Technologies 4(2), pp. 558-582.

Hearst, M. A. 1992. *Automatic Acquisition of Hyponyms from Large Text Corpora.* In Proc. of the 14th conf. on Computational Linguistics, pp. 539-545.

Heylen, K., Peirsman, Y., Geeraerts, D. and Speelman, D. 2008. *Modelling Word Similarity: an Evaluation of Automatic Synonymy Extraction Algorithms.* In Proc. of the 6th LREC.

Hindle, D. 1990. *Noun Classification from Predicate-Argument Structures.* In Proc. of the 28th meeting of ACL, pp. 268-275.

Inkpen, D. 2007. *A Statistical Model for Near-Synonym choice.* In ACM Trans. Speech Lang. Process. 4(1), 1-17.

Kiefer, J. 1953. *Sequential Minimax Search for a Maximum.* In Proc. of the American Mathematical Society 4, pp. 502-506.

Lin, D. 1998. *Automatic Retrieval and Clustering of Similar Words.* In Proc. of the 17th Int. Conf. on Computational Linguistics, pp. 768-774.

Maas, D., Rosener, Ch., and Theofilidis, A. 2009. *Morphosyntactic and Semantic Analysis of Text: The MPRO Tagging Procedure.* Workshop on systems and frameworks for computational morphology. Springer., pp. 76-87.

Maynard, D. F. A. and Peters, W. 2009. *Using lexico-syntactic Ontology Design Patterns for Ontology Creation and Population.* In Proc. of the Workshop on Ontology Patterns.

Moraliyski, R., and Dias, G. 2007. *One Sense per Discourse for Synonym Detection*. In Proc. of the Int. Conf. On Recent Advances in NLP, Bulgaria, pp. 383-387.

Muller, P., Hathout, N. and Gaume, B. 2006. *Synonym Extraction Using a Semantic Distance on a Dictionary*. In Proc. of the 1st Workshop on Graph-Based Methods for NLP, pp. 65-72.

Needleman, S. B. and Wunsch, C. D. 1970. *A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of two Proteins*. In Journal of Molecular Biology 48(3), pp. 443-453.

Ohshima, H. and Tanaka, K. 2009. *Real Time Extraction of Related Terms by Bi-directional lexico-syntactic Patterns from the Web*. In Proc. of the 3rd Int. Conf. on Ubiquitous Information Management and Communication, pp. 441-449.

Riloff, E. 1993. *Automatically Constructing a Dictionary for Information Extraction Tasks*. In Proc. of the 11th Nat. Conf. on Artificial Intelligence, pp. 811-816.

Roark, B. and Charniak, E. 1998. *Noun-phrase Co-occurrence Statistics for Semiautomatic Semantic Lexicon Construction*. In Proc. of the 17th Int. Conf. on Computational Linguistics, pp. 1110-1116.

Smith, T. and Waterman, M. 1981. *Identification of Common Molecular Subsequences*. In Journal of Molecular Biology 147, pp. 195-197.

Sowa, J. F. and Siekmann, J. H. 1994. *Conceptual Structures: Current Practices*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Stevenson, M. and Greenwood, M. 2006. *Comparing Information Extraction Pattern Models*. In Proc. of the Workshop on Information Extraction Beyond the Document, ACL, pp. 29-35.

Terra, E. and Clarke, C. 2003. *Frequency Estimates for Statistical Word Similarity Measures*. In Proc. of HTL/NAACL 2003, pp. 165-172.

Turney, P. D. 2001. *Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL*. Lecture Notes in Computer Science, 2167, pp. 491-502.

Turney, P. D., Littman, M. L., Bigham, J. and Shnayder, V. 2003. *Combining Independent Modules in Lexical Multiple-choice Problems*. In Recent Advances in NLP III: Selected Papers, pp. 101-110.

Turney, P. D. 2008. *A Uniform Approach to Analogies, Synonyms, Antonyms and Associations*. In Proc .of the 22nd Int. Conf. on Computational Linguistics, pp. 905-912.

Van der Plas, L. and Tiedemann, J. 2006. *Finding Synonyms Using Automatic Word Alignment and Measures of Distributional Similarity*. In Proc. of the 21st Int. Conf. on Computational Linguistics, pp. 866-873.

Wei, X., Peng, F., Tseng, H., Lu, Y. and Dumoulin, B. 2009. *Context Sensitive Synonym Discovery for Web Search Queries*. In Proc. of the 18th ACM conference on Information and Knowledge Management, pp. 1585-1588.

Wu, H. and Zhou, M. 2003. *Optimizing Synonym Extraction Using Monolingual and Bilingual Resources*. In Proc. of the 2nd Int. Workshop on Paraphrasing, pp. 72-79.

Finding the Storyteller: Automatic Spoiler Tagging using Linguistic Cues

Sheng Guo
Department of Computer Science
Virginia Tech
guos@cs.vt.edu

Naren Ramakrishnan
Department of Computer Science
Virginia Tech
naren@cs.vt.edu

Abstract

Given a movie comment, does it contain a spoiler? A spoiler is a comment that, when disclosed, would ruin a surprise or reveal an important plot detail. We study automatic methods to detect comments and reviews that contain spoilers and apply them to reviews from the IMDB (Internet Movie Database) website. We develop topic models, based on Latent Dirichlet Allocation (LDA), but using linguistic dependency information in place of simple features from bag of words (BOW) representations. Experimental results demonstrate the effectiveness of our technique over four movie-comment datasets of different scales.

1 Introduction

In everyday parlance, the notion of 'spoilers' refers to information, such as a movie plot, whose advance revelation destroys the enjoyment of the consumer. For instance, consider the movie *Derailed* which features Clive Owen and Jennifer Aniston. In the script, Owen is married and meets Aniston on a train during his daily commute to work. The two of them begin an affair. The adultery is noticed by some inscrupulous people who proceed to blackmail Owen and Aniston. To experience a spoiler, consider this comment from *imdb.com*:

> I can understand why Aniston wanted to do this role, since she gets to play majorly against type (as the supposedly 'nice' girl who's really - oh no! - part of the scam), but I'm at a loss to figure out what Clive Owen is doing in this sub-par, unoriginal, ugly and overly violent excuse for a thriller.

i.e., we learn that Aniston's character is actually a not-so-nice person who woos married men for later blackmail, and thus a very suspenseful piece of information is revealed. Automatic ways to detect spoilers are crucial in large sites that host reviews and opinions.

Arguably, what constitutes a spoiler is inherently a subjective assessment and, for movies/books with intricate storylines, some comments are likely to contain more spoilers than others. We therefore cast the spoiler detection problem as a ranking problem so that comments that are more likely to be spoilers are to be ranked higher than others. In particular, we rank user comments w.r.t. (i.e., given) the movie's synopsis which, according to *imdb*, is '[a detailed description of the movie, including spoilers, so that users who haven't seen a movie can read anything about the title]'.

Our contributions are three fold. (i) We formulate the novel task of spoiler detection in reviews and cast it as ranking user comments against a synopsis. We demonstrate how simple bag-of-words (BOW) representations need to be augmented with linguistic cues in order to satisfactorily detect spoilers. (ii) We showcase the ability of dependency parses to extract discriminatory linguistic cues that can distinguish spoilers from non-spoilers. We utilize an LDA-based model (Wei and Croft, 2006) to probabilistically rank spoilers. Our approach does not require manual tagging of positive and negative examples – an advantage that is crucial to large scale implementation. (iii) We conduct a detailed experimental evaluation with *imdb* to assess the effectiveness of our framework. Using manually tagged com-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 412–420,
Beijing, August 2010

ments for four diverse movies and suitably configured design choices, we evaluate a total of 12 ranking strategies.

2 LDA

Probabilistic topic modeling has attracted significant attention with techniques such as probabilistic latent semantic analysis (PLSA) (Hofmann, 1999) and LDA (Blei et al., 2003; Griffiths and Steyvers, 2004; Heinrich, 2008; Steyvers and Griffiths, 2007). We discuss LDA in detail due to its centrality to our proposed techniques. As a generative model, LDA describes how text could be generated from a latent set of variables denoting topics. Each document is modeled as a mixture of topics, and topics are modeled as multinomial distributions on words.

An unlabeled training corpus can be used to estimate an LDA model. Many inference methods have been proposed, e.g., variational methods (Blei et al., 2003), expectation propagation (Griffiths and Steyvers, 2004), Gibbs sampling (Griffiths and Steyvers, 2004), and a collapsed variational Bayesian inference method (Teh et al., 2007). Gibbs sampling, as a specific form of Markov chain Monte Carlo (MCMC), is a popular method for estimating LDA models. After an LDA model is estimated, it can be used in a very versatile manner: to analyze new documents, for inference tasks, or for retrieval/comparison functions. For instance, we can calculate the probability that a given word appears in a document conditioned on other words. Furthermore, two kinds of similarities can be assessed: between documents and between words (Steyvers and Griffiths, 2007). The similarity between two documents can also be used to retrieve documents relevant to a query document (Heinrich, 2008). Yet another application is to use LDA as a dimensionality reduction tool for text classification (Blei et al., 2003).

To improve LDA's expressiveness, we can relax the bag-of-words assumption and plug in more sophisticated topic models (Griffiths et al., 2005; Griffiths et al., 2007; Wallach, 2006; Wallach, 2008; Wang and Mccallum, 2005; Wang et al., 2007). sLDA (supervised LDA), as a statistical model of labeled collections, focuses on the prediction problem (Blei and Mcauliffe, 2007). The correlated topic model (CTM) (Blei and Lafferty, 2007) addresses plain LDA's inability to model topic correlation. The author-topic model (AT) (Steyvers et al., 2004) considers not only topics but also authors of the documents, and models documents as if they were generated by a two-stage stochastic process.

3 LDA-based spoiler ranking

3.1 Methods

Based on the fact that a spoiler should be topically close to the synopsis, we propose three methods to solve the spoiler ranking problem. The first two use LDA as a preprocessing stage, whereas the third requires positive training data.

Predictive perplexity: Our first method is motivated by the use of LDA-based predictive perplexity (PP) for collaborative filtering (Blei et al., 2003). Here, the PP metric is evaluated over a fixed test dataset in order to empirically compare LDA with other models (pLSI, mixture of unigrams). In our work, we view documents as analogous to users, and words inside documents as analogous to movies. Given a group of known words, we predict the other group of unkown words. We can either calculate the predictive perplexity value from each movie comment Com to the unique synopsis (PP1), or from the synopsis Syn to each comment (PP2).

$$PP1(Syn, \mathbf{w}_{com}) = exp\{-\frac{\sum_{d=1}^{M_{syn}} \log p(w_d|\mathbf{w}_{com})}{M_{syn}}\}$$

$$PP2(Com, \mathbf{w}_{syn}) = exp\{-\frac{\sum_{d=1}^{M_{com}} \log p(w_d|\mathbf{w}_{syn})}{M_{com}}\}$$

In the equations above, $p(w_d|\mathbf{w}_{com})$ and $p(w_d|\mathbf{w}_{syn})$ are the probabilities to generate the word (w_d) from a group of observed words $\mathbf{w}_{obs}$ (either a comment $\mathbf{w}_{com}$ or a synopsis $\mathbf{w}_{syn}$). $p(w|\mathbf{w}_{obs}) = \int \sum_z p(w|z)p(z|\theta)p(\theta|\mathbf{w}_{obs})d\theta$ M_{com} or M_{syn} is the length of a comment or a synopsis. Notice that $p(\theta|\mathbf{w}_{obs})$ can be easily calculated after estimating LDA model by Gibbs sampling. It is also discussed as "predictive likelihood ranking" in (Heinrich, 2008).

Symmetrized KL-divergence: Since documents are modeled as mixtures of topics in LDA, we can calculate the similarity between synopsis and comment by measuring their topic distributions' similarity. We adopt the widely-used symmetrized Kullback Leibler (KL) divergence (Heinrich, 2008; Steyvers and Griffiths, 2007) to measure the difference between the two documents' topic distributions, $sKL(Syn, Com) = \frac{1}{2}[D_{KL}(Syn \| Com) + D_{KL}(Com \| Syn)]$ where $D_{KL}(p\|q) = \sum_{j=1}^{T} p_j \log_2 \frac{p_j}{q_j}$

LPU: Viewing the spoiler ranking problem as a retrieval task given the (long) query synopsis, we also consider the LPU (Learning from Positive and Unlabeled Data) method (Liu et al., 2003). We apply LPU as if the comment collection was the unlabeled dataset, and the synopsis together with few obvious spoiler comments as the positive training data.

3.2 Dependency Parsing

LDA, as a topic model, is widely used as a clustering method and dimensionality reduction tool. It models text as a mixture of topics. However, topics extracted by LDA are not necessarily the same topics as judged by humans since the definition of topic is very subjective. For instance, when conducting sentimental polarity analysis, we hope that topics are clusters concerning one certain kind of subjective sentiment. But for other purposes, we may desire topics focusing on broad 'plots.' Since LDA merely processes a collection according to the statistical distribution of words, its results might not fit either of these two cases mentioned above.

In a basic topic model (section 3.1), neither the order of a sequence of words nor the semantic connections between two words affect the probabilistic modeling. Documents are generated only based on a BOW assumption. However, word order information is very important for most text-related tasks, and simply discarding the order information is inappropriate. Significant work has gone in to address this problem. Griffiths et al. use order information by incorporating collocations (Griffiths et al., 2005; Griffiths et al., 2007). They give an example of the collocation "*united kingdom*", which is ideally treated as a single chunk than two independent words. However, this model can only be used to capture collocations involving sequential terms. Their extended model (Griffiths et al., 2007) integrates topics and syntax, and identifies syntactic classes of words based on their distribution. More sophisticated models exist (Wallach, 2006; Wang and Mccallum, 2005; Wang et al., 2007; Wallach, 2008) but all of them are focused on solving linguistic analysis tasks using topic models. In this paper, however, our focus is on utilizing dependency information as a preprocessing step to help improve the accuracy of LDA models.

In more detail, we utilize dependency parsing to breakup sentences and treat parses as independent 'virtual words,' to be added to the original BOW-based LDA model. In our experiments we employ the Stanford typed dependency parser [1] (Marneffe et al., 2006) as our parsing tool. We use collapsed typed dependencies (a.k.a. grammatical relations) to form the virtual words. However, we do not incorporate all the dependencies. We only retain dependencies whose terms have the part-of-speech tags such as "*NN*", "*VB*", "*JJ*", "*PRP*" and "*RB*"[2], since these terms have strong plot meaning, and are close to the movie topic. Fig. 2 shows a typical parsing result from one sample sentence. This sentence is taken from a review of *Unbreakable*.

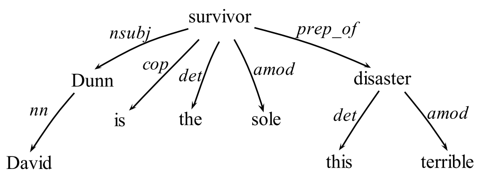


Figure 2: Dependency parse of "David Dunn is the sole survivor of this terrible disaster".

Consider Fig. 1, which depicts five sample sentences all containing two words: "*Dunn*" and "*survivor*". Although these sentences appear different, these two words above refer to the same individual. By treating dependencies as virtual words, we can easily integrate these plot-related relations into an LDA model. Notice that among these five sentences, the grammatical relations between these two words are different: in the fourth sentence, "*survivor*" serves as an appositional modifier of the term "*Dunn*"(appos), whereas in

[1] http://nlp.stanford.edu/software, V1.6

[2] In the implementation, we actually considered all the POS tags with these five tags as prefix, such as "*NNS*", "*VBN*", etc.

David Dunn is the sole survivor of this terrible disaster.

nsubj

David Dunn (Bruce Willis) is the only survivor in a horrific train trash.

nsubj

David Dunn, a man caught in what appears to be a loveless, deteriorating marriage, is the sole survivor of a Philadelphia train wreck.

nsubj

In this Bruce Willis plays David Dunn, the sole survivor of a passenger train accident.

appos

Then the story moves to security guard David Dunn (Bruce Willis) miraculously being the lone survivor of a mile-long train crash (that you find out later was not accidental), and with no injuries what-so-ever.

nsubj

Figure 1: Four sentences with the same topical connection between "Dunn" and "survivor". We integrate this relation into LDA by treating it as a virtual word "Dunn-survivor."

other sentences, "*Dunn*" serves as the nominal subject of "*survivor*"(nsubj). What is important to note is that the surface distances between these given words in different sentences vary a lot. By utilizing dependency parsing, we can capture the semantic connection which is physically separated by even as much as 15 words, as in the third sentence.

We evaluate *topic drift* among the results from plain LDA. We mainly check whether plain LDA will assign the same topic to those terms that have specific linguistic dependency relations. We only consider the following four types of dependencies for evaluation[3]:

- Relations with two noun terms: <NN, NN>, such as "*appos*", "*nn*", "*abbrev*" etc.;
- Relations with one noun and one adjective: <NN, JJ>, like "*amod*";
- Relations with one noun and one verb: <NN, VB>, such as "*agent*", "*dobj*", etc.;
- Relations with only one noun: <NN, *>, which is the relaxed version of <NN, NN>;

We experimented with different pre-set topic numbers (500, 50, and 2) and conducted experiments on four different movie comment collections with LDA analysis. Table 1 shows that <NN, NN> dependency has the highest chance to be topic-matched[4] than other relations. However, all dependencies have very low percentage to be topic-matched, and with a topic number of 2, there remained a significant amount of unmatched <NN, NN> dependencies, demonstrating that simply doing plain LDA may not capture the plot "topic" as we desire.

Observing the results above, each method from section 3.1 (PP1, PP2, sKL and LPU) can be extended by: (1) using BOW-based words, (2) using only dependency-based words, or (3) using a mix of BOW and dependency (dependencies as virtual words). This induces 12 different ranking strategies.

Table 1: Topic match analysis for plain LDA (Each entry is the ratio of topic-matched dependencies to all dependencies)

topic number = 500				
Movie Name	<NN, NN>	<NN, JJ>	<NN, VB>	<NN, *>
Unbreakable	772/3024	412/4411	870/19498	5672/61251
Blood Diamond	441/1775	83/553	80/1012	609/3496
Shooter	242/1846	42/1098	114/2150	1237/15793
Role Models	409/2978	60/1396	76/2529	559/7276
topic number = 50				
Movie Name	<NN, NN>	<NN, JJ>	<NN, VB>	<NN, *>
Unbreakable	1326/3024	953/4411	3354/19498	14067/61251
Blood Diamond	806/1775	151/553	210/1012	1194/3496
Shooter	584/1846	204/1098	392/2150	3435/15793
Role Models	1156/2978	190/1396	309/2529	1702/7276
topic number = 2				
Movie Name	<NN, NN>	<NN, JJ>	<NN, VB>	<NN, *>
Unbreakable	2379/3024	3106/4411	13606/19498	43876/61251
Blood Diamond	1391/1775	404/553	761/1012	2668/3496
Shooter	1403/1846	768/1098	1485/2150	11008/15793
Role Models	2185/2978	908/1396	1573/2529	4920/7276

[3]Here we use <NN, JJ> to express relations having NN and JJ terms, but not necessarily in that order. Also, NN represents all tags related with nouns in the Penn Treebank Tagset, such as NNS. This applies to all the four expressions here.

[4]When both the left term and the right term of a dependency share the same topic, the relation is topic-matched.

Table 2: Some examples of incorrect spoiler tagging in IMDb (italicized sentences are spoilers).

No.	Tag by IMDb	Comment in IMDb
1	Spoiler	The whole film is somewhat slow and it would've been possible to add more action scenes. Even though I liked it very much (6.8/10) I think it is less impressive than "The Sixth Sense" (8.0/10). I would like to be more specific with each scene but it will turn this comment into a spoiler so I will leave it there. I recommend you to see the movie if you come from the basic Sci-Fi generation, otherwise you may feel uncomfortable with it. Anyway once upon a time you were a kid in wonderland and everything was possible. [tt0217869]
2	Spoiler	This is one of the rare masterpiece that never got the respect it deserved because people were expecting sixth sense part 2.Sixth sense was a great film but this is M.N. Shyamalan's best work till date. This is easily one of my top 10 films of all time. Excellent acting, direction, score, cinematography and mood. This movie will hold you in awe from start to finish and any student of cinema would tell what a piece of art this film is. The cast is phenomenal, right from bruce willis to sam jackson and penn , everyone is spectacular in their roles and they make u realise that you do not need loud dramatic moments to create an impact, going slow and subtle is the trick here. This is not a thriller, it's a realistic superhero film. [tt0217869]
3	Spoiler	I can't believe this movie gets a higher rating than the village. OK, after thinking about it, i get the story of unbreakable and i understand what it's trying to say. I do think the plot and the idea is captivating and interesting. Having said that, i don't think the director did anything to make this movie captivating nor interesting. It seemed to try too hard to make this movie a riddle for the audience to solve. The pace was slow at the beginning and ended just as it was getting faster. I remember going out of the cinema, feeling frustrated and confused. it's not until i thoroughly thought about it that i understood the plot. I believe a good movie should engaged the audience and be cleverly suspenseful without confusing the audience too much. Unbreakable tried to be that but failed miserably. 2 out of 10, see the village instead. [tt0217869]
4	Spoiler	This movie touched me in ways I have trouble expressing, and brings forth a message one truly need to take seriously! I was moved, and the ending brought a tear to my eye, as well as a constant two-minute shiver down my spine. It shows how our western way of life influence the lives of thousands of innocents, in a not-so-positive way. Conflict diamonds, as theme this movie debates, are just one of them. Think of Nike, oil, and so on. We continually exploit "lesser developed" nations for our own benefit, leaving a trail of destruction, sorrow, and broken backs in our trail. I, for one, will be more attentive as to what products I purchase in the future, that's for sure. [tt0450259]
5	Non-spoiler	... But the movie takes a while to get to the point. ***"Mr. Glass" has caused lots of mass tragedies in order to find the UNBREAKABLE person. Thus, he is both a mentor and a MONSTER***. ... [tt0217869]
6	Non-spoiler	... This film is about a sniper who loses his best friend while on a shooting mission. A few years later, he is now retired and living in a woodland with his do. Then he is visited by the military to plan an assassination of the president. The shot is fired. ***Unfortunately he is set up to being the shooter and is hunted by cops everywhere. He must find out why he has been set up and also try and stop the real killers***. ... [tt0822854]

4 Experimental Results

4.1 Data preparation

IMDb boasts a collection of more than 203,000 movies (from 1999 to 2009), and the number of comments and reviews for these movies number nearly 970,000. For those movies with synopsis provided by IMDb, the average length of their synopses is about 2422 characters[5]. Our experimental setup, for evaluation purposes, requires some amount of labeled data. We choose four movies from IMDb, together with 2148 comments. As we can see in Table 3, these four movies have different sizes of comment sets: the movie "Unbreakable" (2000) has more than 1000 comments, whereas the movie "Role Models" (2008) has only 123 comments.

Table 3: Evaluation dataset about four movies with different numbers of comments.

Movie Name	IMDB ID	#Comments	#Spoilers
Unbreakable	tt0217869	1219	205
Blood Diamond	tt0450259	538	147
Shooter	tt0822854	268	73
Role Models	tt0430922	123	39

We labeled all the 2148 comments for these four movies manually, and as Table 3 shows, about 20% of each movie's comments are spoilers. Our labeling result is a little different from the current labeling in IMDb: among the 2148 comments, although 1659 comments have the same labels with IMDb, the other 489 are different (205 are treated as spoilers by IMDb but non-spoilers by us; vice versa with 284) The current labeling system in IMDb is very coarse: as shown in Table 2, the first four rows of comments are labeled as spoilers by IMDb, but actually they are not. The last two rows of comments are ignored by IMDb; however, they do expose the plots about the twisting ends.

After crawling all the comments of these four movies, we performed sentence chunking using the LingPipe toolkit and obtained 356 sentences for the four movies' synopses, and 26964 sentences for all the comments of these four movies. These sentences were parsed to extract dependency information: we obtained 5655 dependencies for all synopsis sentences and 448170 dependencies for all comment sentences. From these, we only retain those dependencies that have at least one noun term in either left side or the right side. For measures which require the dependency information, the dependencies are re-organized and treated as a new term planted in the text.

[5]Those movies without synopsis are not included.

4.2 Experiments

4.2.1 Topic number analysis

One of the shortcomings of LDA-based methods is that they require setting a number of topics in advance. Numerous ways have been proposed to handle this problem (Blei et al., 2004; Blei et al., 2003; Griffiths and Steyvers, 2004; Griffiths et al., 2007; Heinrich, 2008; Steyvers and Griffiths, 2007; Teh et al., 2006). Perplexity, which is widely used in the language modeling community, is also used to predict the best number of topics. It is a measure of how well the model fits the unseen documents, and is calculated as average per-word held-out likelihood. The lower the perplexity is, the better the model is, and therefore, the number of topic is specified as the one leading to the best performance. Griffiths and Steyvers (Griffiths and Steyvers, 2004) also discuss the standard Bayesian method which computes the posterior probability of different models given the observed data. Another method from non-parametric Bayesian statistics automatically helps choose the appropriate number of topics, with flexibility to still choose hyper-parameters (Blei et al., 2004; Teh et al., 2006). Although the debate of choosing an appropriate number of topics continues (Boyd-Graber et al., 2009), we utilized the classic perplexity method in our work. Heinrich (Heinrich, 2008) demonstrated that perplexity can be calculated by:

$$P(\tilde{\mathcal{W}}|\mathcal{M}) = \prod_{m=1}^{M} p(\tilde{\vec{w}}_m|\mathcal{M})^{-\frac{1}{N}} = exp\{-\frac{\sum_{m=1}^{M} \log p(\tilde{\vec{w}}_m|\mathcal{M})}{\sum_{m=1}^{M} N_m}\}$$

We chose different topic numbers and calculated the perplexity value for the 20% held-out comments. A good number of topics was found to be between 200 and 600 for both Bow-based strategy and Bow+Dependency strategy, and is also affected by the size of movie comment collections. (We used 0.1 as the document topic prior, and 0.01 as the topic word prior.)

4.2.2 LDA analysis process

As discussed earlier, our task is to rank all the comments according to their possibilities of being a spoiler. We primarily used four methods to do the ranking: PP1, PP2, sKL, and the LPU method. For each method, we tried the basic model using "bag-of-words", and the model using dependency parse information (only), and also with both BOW and dependency information mixed. We utilize LingPipe LDA clustering component which uses Gibbs sampling.

Among the four methods studied here, PP1, PP2 and sKL are based on LDA preprocessing. After obtaining the topic-word distribution and the posterior distributions for topics in each document, the PP1 and PP2 metrics can be easily calculated. The symmetrized KL divergence between each pair of synopsis and comment is calculated by comparing their topic distributions. LPU method, as a text classifier, requires a set of positive training data. We selected those comments which contain terms or phrases as strong hint of spoiler (using a list of 20 phrases as the filter, such as "spoiler alert", "spoiler ahead", etc). These spoiler comments together with the synopsis, are treated as the positive training data. We then utilized LPU to label each comment with a real number for ranking.

4.3 Evaluation

To evaluate the ranking effects of the 12 strategies, we plot *n*-best precision and recall graphs, which are widely used for assessing collocation measures (Evert and Krenn, 2001; Pecina and Schlesinger, 2006). Fig. 3 visualizes the precision-recall graphs from 12 different measures for the four movie comment collections. The *x*-axis represents the proportion of the ranking list, while the *y*-axis depicts the corresponding precision or recall value. The upper part of the figure is the result for the movie which contains more than 1000 comments, while the bottom part demonstrates the result for the relatively small comment collection. The *n*-best evaluation shows that for all the four movie comment collections, PP1_mix and PP1 perform significantly better than the other methods, and the dependency information helps to increase the accuracy significantly, especially for the larger size collection. The LPU method, though using part of the positive training data, did not perform very well. The reason could be that although some of the users put the warning phrases (like "spoiler alert") ahead of their comments, the comment might contain only indirect plot-revealing information. This also reflects that a spoiler tagging method by us-

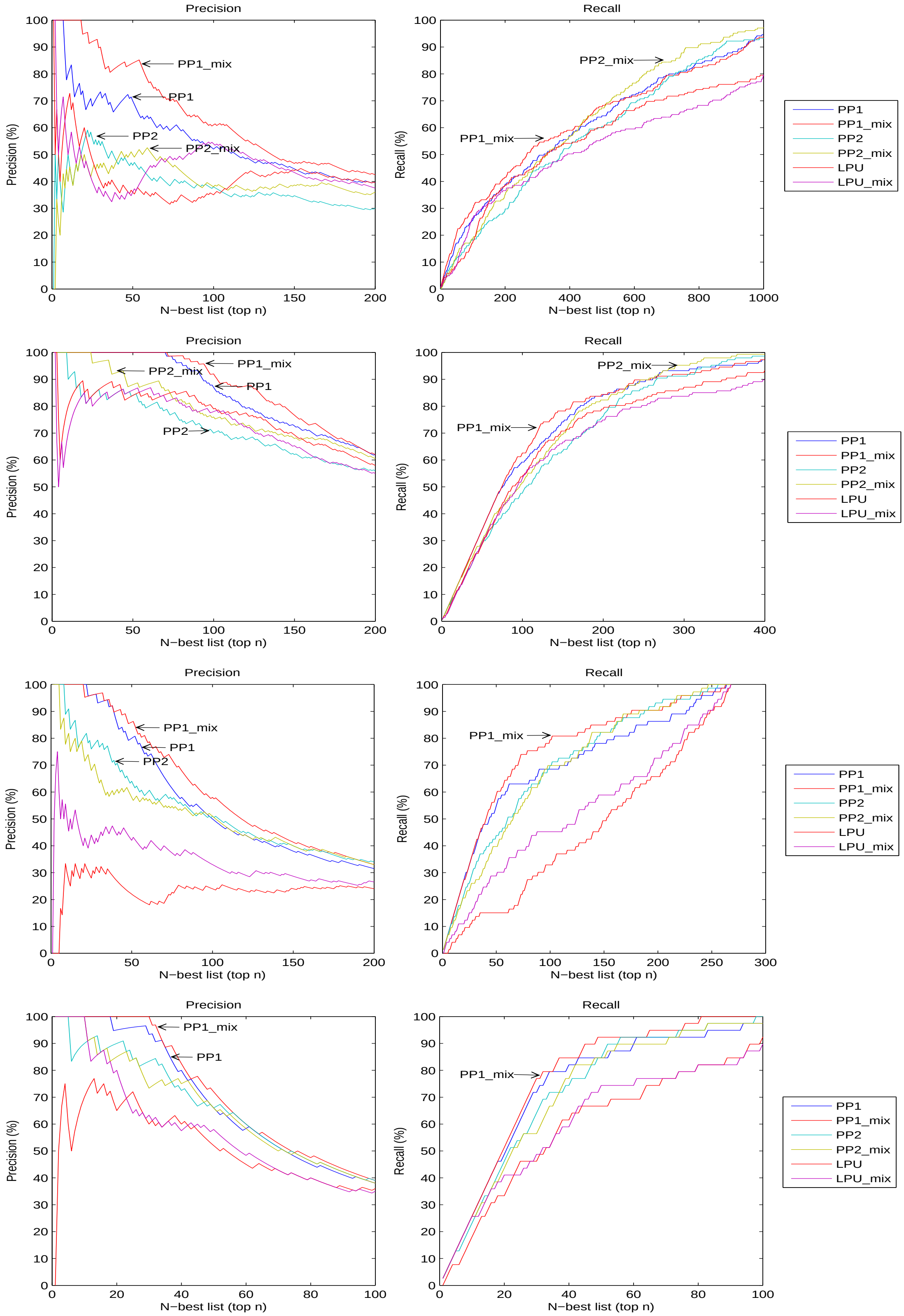


Figure 3: N-best(top *n*th) evaluation (Burnin period = 100): comparison of precision-recall for different methods on four movie comment collections. The PP1 method with BOW and dependency information mixed performs the best among all the measures. Other six methods such as dependency only and KL-based which do not give good performance are ignored in this figure to make it readable. Full comparison is available at: http://sites.google.com/site/ldaspoiler/

ing only keywords typically will not work. Finally, the approach to directly calculating the symmetrized KL divergence seems to be not suitable, either.

4.4 LDA iteration analysis

We also compared the *average precision* values and *normalized discounted cumulative gain* (nDCG) values (Croft et al., 2009; Järvelin and Kekäläinen, 2002) of the ranking results with different parameters for Gibbs sampling, such as burnin period and sample size. Average precision is calculated by averaging the precision values from the ranking positions where a valid spoiler is found, and the nDCG value for the top-p list is calculated as $nDCG_p = \frac{DCG_p}{IDCG} \cdot DCG_p$ is defined as: $DCG_p = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2 i}$ where rel_i is 1 when the *i*-th comment in the list is judged as a real spoiler, and 0, otherwise. IDCG denotes the maximum possible DCG value when all the real spoilers are ranked at the top (*perfect ranking*) (Järvelin and Kekäläinen, 2002).

Table 4: Comparison of ranking by PP_mix using different parameters for Gibbs sampling (analyzed on the top 150 ranking lists, and the values in the table are the mean of the accuracy from four movie comment collections).

Burnin	<S=100; Lag=2>		<S=10; Lag=2>		<S=1; Lag=2>	
	AvgP (%)	nDCG	AvgP (%)	nDCG	AvgP (%)	nDCG
400	80.85	0.951	78.2	0.938	78.1	0.94
200	80.95	0.951	80.5	0.948	79.1	0.94
100	87.25	0.974	80.2	0.943	82.4	0.96
50	81.5	0.958	79.5	0.942	80.0	0.94
10	78.9	0.944	79.5	0.949	75.9	0.92
1	79.4	0.940	79.2	0.952	58.0	0.86

As we can see from Table 4, the accuracy is not affected too much as long as the burin period for the MCMC process is longer than 50 and the sample size retained is larger than 10. In our experiments, we use 100 as the burin parameter, and beyond that, 100 samples were retained with sample lag of 2.

4.5 Representative results

As shown in Table 5, we find that the basic BOW strategy prefers the longer comments whereas the strategy that uses dependency information prefers the shorter ones. Although it is reasonable that a longer comment would have a higher probability of revealing the plot, methods which prefers the longer comments usually leave out the short spoiler comments. By incorporating the dependency information together with the basic BOW, the new method reduces this shortcoming. For instance, consider one short comment for the movie "*Unbreakable* (2000)":

> This is the same formula as Sixth Sense – from the ability to see things other people don't, to the shocking ending. Only this movie is just not plausible – I mean Elijah goes around causing disasters, trying to see if anyone is "Unbreakable" – it's gonna take a lot of disasters because its a big world.

whcih is ranked as the 27th result in the PP1_mix method, whereas the BOW based PP1 method places it at the 398th result in the list. Obviously, this comment reveals the twisting end that it is Elijah who caused the disasters.

Table 5: Comparison of average length of the top-50 comments of 4 movies from 2 strategies.

	Role Models	Shooter	Blood Diamond	Unbreakable
BOW	2162.14	2259.36	2829.86	1389.18
Dependency	1596.14	1232.12	2435.58	1295.72

5 Conclusions and future work

We have introduced the spoiler detection problem and proposed using topic models to rank movie comments according to the extent they reveal the movie's plot. In particular, integrating linguistic cues from dependency information into our topic model significantly improves the ranking accuracy.

In future work, we seek to study schemes which can segment comments to potentially identify the relevant spoiler portion automatically. The automatic labeling idea of (Mei et al., 2007) can also be studied in our framework. Deeper linguistic analysis, such as named entity recognition and semantic role labeling, can also be conducted. In addition, evaluating topic models or choosing the right number of topics using dependency information can be further studied. Finally, integrating the dependency relationships more directly into the probabilistic graphical model is also worthy of study.

References

Blei, David M. and John D. Lafferty. 2007. A correlated topic model of science. *Annals of Applied Statistics*, 1(1):17–35.

Blei, David M. and Jon D. Mcauliffe. 2007. Supervised topic models. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of machine learning research*, 3:993–1022.

Blei, David M., T. Gri, M. Jordan, and J. Tenenbaum. 2004. Hierarchical topic models and the nested chinese restaurant process. In *Proceedings of the 18th Annual Conference on Neural Information Processing Systems*.

Boyd-Graber, Jordan, Jonathan Chang, Sean Gerrish, Chong Wang, and David Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*.

Croft, Bruce, Donald Metzler, and Trevor Strohman. 2009. *Search Engines: Information Retrieval in Practice*. Addison Wesley, 1 edition.

Evert, Stefan and Brigitte Krenn. 2001. Methods for the qualitative evaluation of lexical association measures. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*.

Griffiths, Thomas L. and M. Steyvers. 2004. Finding scientific topics. In *Proceedings of the National Academy of Sciences of the United States of America*, 101 Suppl 1:5228–5235, April.

Griffiths, Thomas L., Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. 2005. Integrating topics and syntax. In *Proceedings of the 19th Annual Conference on Neural Information Processing Systems*.

Griffiths, Thomas L., Mark Steyvers, and Joshua B. Tenenbaum. 2007. Topics in semantic representation. *Psychological Review*, 114(2):211–244, April.

Heinrich, Gregor. 2008. Parameter estimation for text analysis. Technical report, University of Leipzig.

Hofmann, Thomas. 1999. Probabilistic latent semantic analysis. In *Proceedings of 15th Conference on Uncertainty in Artificial Intelligence*.

Järvelin, Kalervo and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446.

Liu, Bing, Yang Dai, Xiaoli Li, Wee Lee, and Philip S. Yu. 2003. Building text classifiers using positive and unlabeled examples. In *Proceedings of the 3rd IEEE International Conference on Data Mining*.

Marneffe, M., B. Maccartney, and C. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*.

Mei, Qiaozhu, Xuehua Shen, and ChengXiang Zhai. 2007. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD conference*.

Pecina, Pavel and Pavel Schlesinger. 2006. Combining association measures for collocation extraction. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*.

Steyvers, Mark and Tom Griffiths, 2007. *Probabilistic Topic Models*. Lawrence Erlbaum Associates.

Steyvers, Mark, Padhraic Smyth, Michal R. Zvi, and Thomas Griffiths. 2004. Probabilistic author-topic models for information discovery. In *Proceedings of the 10th ACM SIGKDD conference*.

Teh, Yee Whye, Jordan, I. Michael, Beal, J. Matthew, Blei, and M. David. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, December.

Teh, Yee W., David Newman, and Max Welling. 2007. A collapsed variational bayesian inference algorithm for latent dirichlet allocation. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*.

Wallach, Hanna M. 2006. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd International Conference on Machine Learning*.

Wallach, Hanna M. 2008. *Structured topic models for language*. Ph.D. thesis, University of Cambridge.

Wang, Xuerui and Andrew Mccallum. 2005. A note on topical n-grams. Technical report, University of Massachusetts Amherst.

Wang, Xuerui, Andrew McCallum, and Xing Wei. 2007. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Proceedings of the 7th IEEE International Conference on Data Mining*.

Wei, Xing and Bruce W. Croft. 2006. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference*.

Detection of Simple Plagiarism in Computer Science Papers

Yaakov HaCohen-Kerner
Department of Computer Science, Jerusalem College of Technology (Machon Lev)
kerner@jct.ac.il

Aharon Tayeb
Department of Computer Science, Jerusalem College of Technology (Machon Lev)
aharontayeb@gmail.com

Natan Ben-Dror
Department of Computer Science, Jerusalem College of Technology (Machon Lev)
bd.natan@gmail.com

Abstract

Plagiarism is the use of the language and thoughts of another work and the representation of them as one's own original work. Various levels of plagiarism exist in many domains in general and in academic papers in particular. Therefore, diverse efforts are taken to automatically identify plagiarism. In this research, we developed software capable of simple plagiarism detection. We have built a corpus (C) containing 10,100 academic papers in computer science written in English and two test sets including papers that were randomly chosen from C. A widespread variety of baseline methods has been developed to identify identical or similar papers. Several methods are novel. The experimental results and their analysis show interesting findings. Some of the novel methods are among the best predictive methods.

1 Introduction

In light of the explosion in the number of available documents, fast and accurate searching for plagiarism is becoming more needed. Identification of identical and similar documents is becoming very important.

Plagiarism is the use of the language and thoughts of another work and the representation of them as one's own original work (Wikipedia, 2010; Library and Information Services, 2010). Plagiarism can be committed by "recycling" other's work as well as by one’s own work (self-plagiarism).

Various levels of plagiarism exist in many domains in general and in academic papers in particular. In addition to the ethical problem, plagiarism in Academics can be illegal if copyright of the previous publication has been transferred to another entity.

It is important to mention, that in many cases similar papers are different versions of the same work, e.g., a technical report, a poster paper, a conference paper, a journal paper and a Ph. D. dissertation.

To avoid any kind of plagiarism, all sources which were used in the completion of a work/research must be mentioned (Library and Information Services, 2010).

Over the last decade, various softwares have been built to automatically identify plagiarism (e.g., Collberg et al. (2005), Sorokina et al. (2006), and Keuskamp and Sliuzas (2007)).

In this research, we developed such a system. This system is planned to deal with simple kinds of plagiarism, e.g., copying of sentences or part of sentences. We have built a corpus that contains academic papers in computer science written in English. Most of the papers are related to the domain research of Natural Language Processing (NLP) and are from the last ten years.

The remainder of this paper is organized as follows: Section 2 gives a background regarding plagiarism. Section 3 overviews researches and systems dealing with detection of plagiarism. Section 4 describes five groups of baseline methods, which have been implemented by us to detect plagiarism. Section 5 presents the experiments that have been performed and their analysis. Section 6 gives an illustrative example. Section 7 concludes and proposes future directions for research.

2 Plagiarism

Plagiarism is defined in the 1995 Random House Compact Unabridged Dictionary as the "use or close imitation of the language and thoughts of another author and the representation of them as one's own original work."

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 421–429, Beijing, August 2010

Self-plagiarism is the reuse of significant, identical, or nearly identical parts of one's own work without citing the original work. In addition to the ethical issue, this phenomenon can be illegal if copyright of the previous work has been transferred to another entity. Usually, self-plagiarism is considered to be a serious ethical problem in cases where a publication needs to contain an important portion of a new material, such as in academic papers (Wikipedia, 2010).

On the other hand, it is common for researchers to rephrase and republish their research, tailoring it for different academic journals and conference articles, to disseminate their research to the widest possible interested public. However, these researchers must include in each publication a meaningful or an important portion of a new material (Wikipedia, 2010).

There are various classifications for levels of plagiarism. For instance, IEEE (2010) categorized plagiarism into five levels, or degrees, of misconduct, ranging from the most serious (Level One) to the least serious (Level Five):

Level One: The uncredited verbatim copying of a full paper, or the verbatim copying of a major portion (greater than half of the original paper)

Level Two: The uncredited verbatim copying of a large portion (less than half of the original paper).

Level Three: The uncredited verbatim copying of individual elements (e.g., paragraphs, sentences, figures).

Level Four: The uncredited improper paraphrasing of pages or paragraphs.

Level Five: The credited verbatim copying of a major portion of a paper without clear delineation (e.g., quotes or indents).

Loui (2002) handled eight allegations of plagiarism related to students' works. Collberg et al. (2005) proposes eight ranks of plagiarism.

3 Related Research

There are two main attitudes concerning discovery of similar documents: ranking and fingerprinting. Ranking methods are derived from information retrieval (IR) and are widely used in IR systems and Internet search engines. Known ranking methods are the cosine measure, the inner product, and the normalized inner product. Hoad and Zobel (2003) extended the ranking family by defining identity measures, designed for identification of co-derivative documents.

Fingerprinting aims to compare between two documents based on their fingerprints. Fingerprint methods have been used by many previous researches, e.g., Manber (1994). Heintze (1996), Lyo et al. (2001), Hoad and Zobel (2003), and Shivakumar and Garcia-Molina (1996).

3.1 Full Fingerprinting

Given a document, a full fingerprint of the document consists of the set of all the possible sequential substrings of length α in words (a definition that is based on characters is also possible). There are $N-\alpha+1$ such substrings, where N is the length of the document in words. This fingerprinting selects overlapping sub-strings. For instance, if α is 3, this method selects the 3-word phrases that begin at position 0; 1; 2; etc. The size of α is known as the fingerprint granularity. This variable can have a significant impact of the accuracy of fingerprinting (Shivakumar and Garcia-Molina, 1996).

Comparing a document X to a document Y where X's size is $|X|$ and if n is the number of substrings common to both documents then $n/|X|$ is the measure of how much of X is contained in Y.

3.2 Selective Fingerprinting

To decrease the size of a full fingerprint, there are various versions of selective fingerprints.

The simplest kind of selective fingerprinting is the "All substrings selection" described in Hoad and Zobel (2003). This fingerprinting is similar to the full fingerprinting, but it does not select overlapping sub-strings. Rather, it selects all non-overlapping substrings of size α (in words) from the document. For example, if α is 3, this strategy selects the 3-word phrases that begin at position 0; 3; 6; 9; etc.

Heintze (1996) performed various experiments using a fixed number of fingerprints independent of the size of the document and a fixed number of substrings of size α (in characters). The best results were achieved by 1,000 fingerprints with α=50. Another possibility is to work with a fixed proportion of the substrings, so that the size of the selective fingerprint is proportional to the size of the document. The main disadvantage of this possibility is space consumption.

Hoad and Zobel (2003) suggested many additional general types of selective fingerprinting, e.g., positional, frequency-based, and structure-based.

3.3 Additional Similarity Measures

SymetricSimilarity

Monostori1 et al. (2002) defined a measure called SymetricSimilarity as follows:

$SS(X, Y) = |d(X) \cap d(Y)| / |d(X) + d(Y)|$

where X and Y are the two compared documents, d(X) and d(Y) are the number of the fingerprints of X and Y, respectively, and $|d(X) \cap d(Y)|$ is the number of the common fingerprints.

S2 and S3

Bernstein and Zobel (2004) defined several additional similarity measures, such as S2 and S3:

$S2(X, Y) = |d(X) \cap d(Y)| / \min(|d(X)|, |d(Y)|)$

$S3(X, Y) = |d(X) \cap d(Y)| / ((|d(X)| + |d(Y)|)/2)$

where $\min(|d(X)|, |d(Y)|)$ is the minimal number of the fingerprints of X and Y, respectively, and $|d(X) \cap d(Y)|$ is the average number of the fingerprints of X and Y.

Rarest-in-document

The Rarest-in-Document method is one of the frequency-based methods defined by Hoad and Zobel (2003). This method chooses the substrings that produce the rarest substrings with length of k words in the document. This means that all of the substrings must be calculated and sorted according to their frequency in the document, and then the rarest of them are selected. The intuition is that sub-strings, which are less common, are more effective discriminators when comparing documents for similarity.

Anchor methods

Hoad and Zobel (2003) defined anchor methods. These methods are based on specific, predefined strings (called anchors), in the text of the document. The anchors are chosen to be common enough that there is at least one in almost every document, but not so common that the fingerprint becomes very large (Manber, 1994).

Various anchors were used by Hoad and Zobel. The anchors were randomly selected, but extremely common strings such as "th" and "it" were rejected. The 35 2-character anchor method detects all of the documents that were considered as similar by a human expert.

Additional experiments have been applied to identify the optimal size of an anchor. Manber (1994) used 50-character anchors in a collection of over 20,000 "readme" documents, identifying 3,620 sets of identical files and 2,810 sets of similar files. Shivakumar and Garcia-Molina (1996) achieved the best results with one-sentence anchors and Heintze (1996) achieved the best results with 1000-character anchors.

4 Baseline Detection Methods

To find whether there is a plagiarism, novel and old baseline methods have been implemented. These methods can be divided into five groups: full fingerprint methods, selective fingerprint methods, anchor methods, word comparison methods, and combinations of methods.

Full fingerprint methods

All the full fingerprint methods are defined for overlapping substrings of length k in words from the beginning of the document.

1. FF(k) - Full Fingerprints of length k
2. SSF(k) - SymetricSimilarity for Full fingerprints of length k
3. S2F(k) - S2 for Full fingerprints of length k
4. S3F(k) - S3 for Full fingerprints of length k
5. RDF(k) - Rarest-in-Document for Full fingerprints of length k
6. CA - Compare between the abstracts of the two documents using FF(3)

Selective Fingerprint methods

In this research, all the selective fingerprint methods are selective by the sense of non-overlapping substrings of length k in words from the beginning of the document.

7. SF(k) - Selective Fingerprints of length k

8. SSS(k) - SymetricSimilarity for Selective fingerprints of length k
9. S2S(k) - S2 for Selective fingerprints of length k
10. S3S(k) - S3 for Selective fingerprints of length k
11. RDS(k) - Rarest-in-Document for Selective fingerprints of length k

Anchor methods

We decided to work with seventy (N=70) 3-character anchors. Based on these anchors we have defined the following methods:

12. AFW - Anchor First Words - First 3-charcters from each one of the first N words in the tested document
13. AFS - Anchor First Sentences - First 3-charcters from each one of the first N sentences in the tested document
14. AF - most Frequent Anchors - N most frequent 3-charcter prefixes in the tested document
15. AR - Rarest Anchors - N rarest frequent 3-charcter prefixes in the tested document
16. ALW - Anchor Last Words - First 3-charcters from each one of the last N words in the tested document
17. ALS - Anchor Last Sentences - First 3-charcters from each one of the last N sentences in the tested document Word comparisons
18. CR - CompareReferences. This method compares between the titles of the papers included in the references section of the two examined papers.

Combinations of methods

19. CARA- CompareAbstractReferencesAverage. This method returns the average value of CA and CR.
20. CARM - CompareAbstractReferencesMin. This method returns the minimal value between CA and CR.

As mentioned above, Hoad and Zobel (2003) defined anchor methods based on the first/last N sentences/words/3-charcter prefixes in the tested document. As shown in Table 1 and in its analysis, the anchor methods are not successful, probably because they use a small portion of data. Therefore, we decided to implement methods defined for the following portions of the paper: the first third (*first*), the middle third (*middle*), and the last third (*end*) of the paper according to the number of the words in the discussed paper. All the *first*, *middle* and *end* methods use FF(3). These methods were combined with CA or CR. CA was not combined with the *first* methods because the abstract is included in the first part of the paper. CR was not combined with the *last* methods because the references are included in the end part of the paper.

21. CAMA- CompareAbstractMiddleAve. This method returns the average value of CA and FF(3) computed for the middle parts of the two examined papers.
22. CAMM - CompareAbstractMiddleMin. This method returns the minimal value between CA and FF(3) computed for the middle parts of the two examined papers.
23. CAEA - CompareAbstractEndAverage. This method returns the average value of CA and FF(3) computed for the end parts of the two examined papers.
24. CAEM - CompareAbstractEndMin. This method returns the minimal value between CA and FF(3) computed for the end parts of the two examined papers.
25. CRFA - CompareReferencesFirstAverage. This method returns the average value of CR and FF(3) computed for the first parts of the two examined papers.
26. CRFM - CompareReferencesFirstMin. This method returns the minimal value between between CR and FF(3) computed for the first parts of the two examined papers.
27. CRMA - CompareReferencesMiddleAverage. This method returns the average value of CR and FF(3) computed for the middle parts of the two examined papers.
28. CRMM - CompareReferencesMiddleMin. This method returns the minimal value between CR and FF(3) computed for the middle parts of the two examined papers.

To the best of our knowledge, we are the first to implement methods that compare special and important sections in academic papers: abstract and references: CA and CR, and combinations of them. In addition, we implemented new methods defined for the three thirds: the first (F) third, the middle (M) third, and the last (E) third of the paper. These methods were combined with CA and CR in various variants. All in total, we have defined 12 new baseline methods.

5 Experimental Results

5.1 Dataset

As mentioned above, the examined dataset includes 10,100 academic papers in computer science. Most of the papers are related to NLP and are from the last ten years. Most of the papers were downloaded from http://www.aclweb.org/anthology/.

These documents include 52,909,234 words that are contained in 3,722,766 sentences. Each document includes in average 5,262 words. The maximum number of words in a document is 28,758. The minimum number of words in a document is 305.

The original PDF files were downloaded using IDM - Internet Download Manager (http://www.internetdownloadmanager.com/). Then we convert them to TXT files using ghostscript (http://pages.cs.wisc.edu/~ghost/). Many PDF files were not papers and many others were converted to gibberish files. Therefore, the examined corpus contains only 10,100 papers.

5.2 Experiment I

Table 1 presents the results of the 38 implemented methods regarding the corpus of 10,100 documents. The test set includes 100 papers that were randomly chosen from the examined dataset. For each tested document, all the other 10,099 documents were compared using the various baseline methods.

The IDN, VHS, HS, MS columns present the number of the document pairs that found as identical, very high similar, high similar, and medium similar to the 100 tested documents, respectively. The IDN, VHS, HS, MS levels were granted to document pairs that got the following similarity values: 100%, [80%, 100%), [60%, 80%), and [40%, 60%), respectively.

The first left column indicates a simple ordinal number. The second left column indicates the serial number of the baseline method (Section 4) and the number in parentheses indicates the number of the chosen words (3 or 4) to be included in each substring.

On the one hand, the anchor methods (# 12-17) tried on the interval of 70-500 anchors report on relatively high numbers of suspicious document pairs, especially at the MS level. According to our expert, these high numbers are rather exaggerated. The reason for this finding might be that such fix numbers of anchors are not for detection of similar papers in various degrees of similarity.

#	#(k)	Method	IDN	VHS	HS	MS
1	1(3)	FF(3)	9	0	2	1
2	1(4)	FF(4)	9	0	1	1
3	2(3)	SSF(3)	0	0	0	9
4	2(4)	SSF(4)	0	0	0	9
5	3(3)	S2F(3)	9	0	2	2
6	3(4)	S2F(4)	9	0	1	1
7	4(3)	S3F(3)	0	0	9	0
8	4(4)	S3F(4)	0	0	9	0
9	5(3)	RDF(3)	1	5	1	3
10	5(4)	RDF(4)	1	6	0	3
11	6	CA	9	0	1	0
12	7(3)	SF(3)	9	0	0	1
13	7(4)	SF(4)	9	0	0	1
14	8(3)	SSS(3)	0	0	0	9
15	8(4)	SSS(4)	0	0	0	9
16	9(3)	S2S(3)	9	0	0	1
17	9(4)	S2S(4)	9	0	0	1
18	10(3)	S3S(3)	0	0	9	0
19	10(4)	S3S(4)	0	0	9	0
20	11(3)	RDS(3)	0	0	0	1
21	11(4)	RDS(4)	0	0	0	0
22	12	AFW	4	6	18	2772
23	13	AFS	6	3	10	708
24	14	AF	6	4	4	313
25	15	AR	4	6	19	2789
26	16	ALW	4	6	9	500
27	17	ALS	4	5	12	704
28	18	CR	9	0	1	3
29	19	CARA	8	2	1	0
30	20	CARM	8	0	2	0
31	21	CAMA	9	0	1	0
32	22	CAMM	9	0	0	1
33	23	CAEA	9	0	1	0
34	24	CAEM	9	0	0	1
35	25	CRFA	8	0	3	0
36	26	CRFM	8	0	2	0
37	27	CRMA	8	0	3	0
38	28	CRMM	8	0	1	1

Table 1. Results of the 38 implemented methods for 100 tested papers.

On the other hand, the SSF(k), S3F(k), S3S(k), RDF(k), and RDS(k) methods report on relatively very low numbers of suspicious document pairs. According to our expert, these numbers are too low. The reason for this finding might be that these methods are quite stringent for detection of similar document pairs.

The full fingerprint methods: FF(k), S2F(k) and the selective fingerprint methods SF(k), and S2S(k) present very similar results, which are reasonable according to our expert. Most of these methods report on 9 IDN, 0 VHS, 0-2 HS, and 1-2 MS document pairs. The full fingerprint methods report on slightly more HS and MS document pairs. According to our expert, these methods are regarded as the best.

Our novel methods: CA and CR also report on 9 IDN, 0 VHS, one HS, and 0 or 3 MS document pairs, respectively. The sum (10-13) of the IDN, VHS, HS and MS document pairs found by the best full and selective fingerprint methods mentioned in the last paragraph is the same sum of the IDN, VHS, HS and MS document pairs found by the CA and CR methods. That is, the CA and CR are very close in their quailty to the best methods. However, the CA and the CR have a clear advantage on the other methods. They check a rather small portion of the papers, and therfore their run time is much more smaller.

On the one hand, CR seems to be better than CA (and even the best selective fingerprint methods SF(k), and S2S(k)) because it reports on more MS document pairs, which means that CR is closer in its quality to the best full fingerprint methods. On the other hand, according to our expert CA is better than CR, since CR has more detection failures.

The combinations of CA and/or CR and/or the methods defined for the three thirds of the papers report on results that are less or equal from the viewpoint of their quality to CA or CR.

Several general conclusions can be drawn from the experimental results as follows:

(1) There are 9 documents (in the examined corpus) that are identical to one of the 100 tested papers. According to our expert, each one of these documents is IDN to a different paper from the 100 tested papers. This means that at least 9% of our random tested papers have IDN files in a corpus that contains 10, 099 files (for each test file).

(2) Several papers that have been found as IDN might be legal copies. For example: (a) by mistake, the same paper might be stored twice at the same conference website or (b) the paper, which is stored in its conference website might also be stored at its author's website.

(3) All the methods that run with two possible values of k (3 or 4 words) present similar results for the two values of k.

(4) FF(3) found as better than FF(4). FF(3) discovers 9 IDN papers, 2 HS papers, and 1 MS paper. These results were approved by a human expert. FF(4) missed one paper. One HS paper identified by FF(3) was identified as MS by FF(4) and one MS paper identified by FF(3) was identified as less than MS by FF(4). Moreover, also for other methods, variants with K=3 were better or equal to those with K=4. The main reason for these findings might be that the variants with K=4 check less substrings because the checks are done for each sentence. Substrings that end at the sequential sentence are not checked. Therefore, it is likely that additional equal substrings from the checked papers are not identified.

(5) S2F(3) discovers one more MS paper compared to FF(3). According to the human expert, the similarity measure of this paper should be less than MS. Therefore, we decided to select FF(3) as the best method.

(6) FF(3)'s run time is very high since it works on overlapping substrings for the whole papers.

(7) Our two novel methods: CA and CR are among the best methods for identification of various levels of plagiarism. As mentioned before, CA was found as a better predictor.

5.3 Selection of Methods and Experiment II

Sixteen methods out of the thirty-eight methods presented in Table 1, were selected for additional experiments. All the methods with k=4, the anchor methods, SSF, S3F, S3S, RDF, and RDS methods were omitted, due to their faulty results (as explained above). The remaining 16 methods (with k=3) are: FF, S2F, S2F, SF, S2S and all our 12 baseline methods: CA, and CR- CRMM.

Table 2 presents the results of these methods regarding the corpus of 10,100 documents. Since we selected less than half of the original methods we allow ourselves to test 1,000 documents instead of 100.

#	Method	IDN	VHS	HS	MS	Time d:h:m
1	FF	38	0	11	5	1:3:57.3
2	S2F	41	1	10	18	32:00.0
3	SF	37	1	1	6	31:12.2
4	S2	38	1	1	14	20:10.8
5	CA	38	1	11	5	09:16.7
6	CR	41	2	11	67	05:57.7
7	CARA	33	2	1	21	31:53.4
8	CARM	30	4	1	5	33:40.1
9	CAMA	38	0	5	6	11:26.5
10	CAMM	38	0	3	4	10:09.8
11	CAEA	38	0	6	7	10:42.1
12	CAEM	38	0	3	4	12:35.3
13	CRFA	32	1	3	25	54:20.7
14	CRFM	30	3	3	6	54:10.0
15	CRMA	33	2	3	25	58:52.2
16	CRMM	30	2	2	5	54:17.7

Table 2. Results of the 16 selected methods for 1,000 tested papers.

Again, according to our expert, FF has been found as the best predictive method. Surprisingly, CA achieved the second best results with one additional VHS paper. 11 HS documents and 5 MS documents have been identified by CA as by FF. The meaning of this finding is that the abstracts in almost all the simple similar documents were not significantly changed. That is, the authors of the non-IDN documents did not invest enough to change their abstracts.

CR indentified 41 documents as identical. The reason for this is probably because 3 additional papers have the same reference section as in 3 other tested papers, although these 3 document pairs are different in other sections. Furthermore, CR reports on relatively high number of suspicious document pairs, especially at the MS level. The meaning of this finding is that the references in many document pairs are not significantly different although these documents have larger differences in other sections. Consequently, combinations with CA achieved better results than combinations with CR.

A very important finding is that the run time of FF was very expensive (one day, 3 hours and 57.3 minutes) compared to the run time of CA (9 hours and 16.7 minutes). In other words, CA achieved almost the same results as FF but more efficiently.

5.4 An Error Analysis

The selected methods presented in Table 2 were analyzed according to the results of FF. Table 3 shows the distributions of false true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), regarding the 10,099 retrieved documents for the 1,000 tested document.

The false positive rate is the proportion in percents of positive test results (i.e., a plagiarism was identified by a baseline function) that are really negative values (i.e., the truth is that there is no plagiarism). The false negative rate is the proportion of negative test results that are really positive values.

#	Method	TP	FP	TN	FN
1	FF	0.534	0	99.465	0
2	S2F	0.524	0.168	99.296	0.010
3	SF	0.425	0.019	99.445	0.108
4	S2	0.435	0.099	99.366	0.099
5	CA	0.534	0.010	99.455	0
6	CR	0.534	0.663	98.801	0
7	CARA	0.386	0.178	99.287	0.148
8	CARM	0.356	0.039	99.425	0.178
9	CAMA	0.475	0	99.465	0.059
10	CAMM	0.445	0	99.465	0.089
11	CAEA	0.485	0.020	99.445	0.049
12	CAEM	0.445	0	99.465	0.089
13	CRFA	0.396	0.207	99.257	0.138
14	CRFM	0.376	0.039	99.425	0.158
15	CRMA	0.405	0.217	99.247	0.128
16	CRMM	0.366	0.020	99.445	0.168

Table 3. Distributions of the various possible statistical results.

FF is the only method that detects all cases of simple plagiarism. According to FF, there are 0.534% true positives. That is, 54 papers out of 10,099 are suspected as plagiarized versions of

54 papers of the 1,000 tested papers. This finding fits the results of FF(3) in Table 2, where there are 38 IDN, 11 HS, and 5 MS.

CA, the second best method has 0% false positives, and 0.01% false negatives, which means that CA identified one suspected plagiarized version that is really a non-plagiarized document. This finding is presented in Table 2, where CA identified 55 suspected plagiarized documents, one more than FF.

CR has 0% false positives, and 0.663% false negatives, which means that CR identified 67 suspected plagiarized versions that are really non-plagiarized documents. This finding is presented in Table 2, where CR identified 121 suspected plagiarized documents, 67 more than FF.

6 Illustrative Example

Due to space limitations, we briefly present an illustrative example of comparison between a couple of papers found as HS (High Similar) according to FF(3), the best detection method.

The tested paper (Snider and Diab, 2006A) contains 4 pages and it was published on June 06. The retrieved paper (Snider and Diab, 2006B) contains 8 pages and it was published a month later. The title of the tested paper is identical to the first eight words of the title of the retrieved paper. The authors of both papers are the same and their names appear in the same order. Most of the abstracts are the same. One of the main differences is the report of other results (probably updated results).

A relatively big portion of the beginning of the Introduction section in both papers is identical. Very similar sentences are found at the beginning of different sections (Section 2 in the 4-page paper and Section 3 in the the 8-page paper).

Many sentences or phrases from the rest of the papers are identical and some are very similar (e.g., addition of 'The' before "verbs are classified" in the abstract of the retrieved paper.

It is imoprtant to point that the authors in their 8-page paper wrote "This paper is an extension of our previous work in Snider and Diab (2006)". This sentence together with the detailed reference prove that the authors cite their previous work as required.

Concerning the references in both papers, at the first glance we found many differences between the two papers. The short paper contains only 7 references while the larger paper contains 14 references. However, a second closer look identifies that 5 out of the 7 references in the shorter paper are found in the reference section of the larger paper. Indeed, regarding the reference sections we did not find HS; but we have to remember that the larger paper include 8 pages twice than the shorter paper and therfore, more references could be included.

7 Conclusions and Future Work

To the best of our knowledge, we are the first to implement the CA and CR methods that compare two basic and important sections in academic papers: the abstract and references, respectively. In addition, we defined combinations of them. Furthermore, we implemented methods defined for the three thirds of the paper. These methods were combined with CA or CR in various variants. All in total, we have defined 12 new baseline methods.

Especially CA and also CR are among the best methods for identification of various levels of plagiarism. In contrast to the best full and selective fingerprint methods, CA and CR check a rather small portion of the papers, and therfore, their run time is much more smaller.

The success of CA and CR teaches us that most documents that are suspected as simple plagiarized papers include abstracts and references, which have not been significantly changed compared to other documents or vice versa.

There is a continuous need for automatic detection of plagiarism due to web influences, and advanced and more complex levels of plagiarism. Therefore, some possible future directions for research are: (1) Developing new kinds of selective fingerprint methods and new combinations of methods to improve detection, (2) Applying this research to larger and/or other corpora, and (3) Dealing with complex kinds of plagiarism, e.g., the use of synonyms, paraphrases, and transpositions of active sentences to passive sentences and vice versa.

References

Bernstein, Y., and Zobel, J., 2004. A Scalable System for Identifying Co-Derivative Documents. In *Proceedings of 11th International Conference on String Processing and Information Retrieval (SPIRE)*, vol. 3246, pp. 55-67.

Bretag, T., and Carapiet, S., 2007. A Preliminary Study to Identify the Extent of Self Plagiarism in Australian Academic Research. *Plagiary*, 2(5), pp. 1-12.

Collberg, C., Kobourov, S., Louie, J., and Slattery, T., 2005. Self-Plagiarism in Computer Science. *Communications of the ACM*, 48(4), pp. 88-94.

Heintze, N., 1996. Scalable Document Fingerprinting. In *Proceedings of the USENIX Workshop on Electronic Commerce*, Oakland California.

Hoad, T. C., and Zobel, J., 2003. Methods for Identifying Versioned and Plagiarised Documents. *Journal of the American Society for Information Science and Technology*, Vol 54(3), pp. 203-215.

IEEE, 2010. Introduction to the Guidelines for Handling Plagiarism Complaints. http://www.ieee.org/publications_standards/publications/rights/plagiarism.html.

Keuskamp, D., and Sliuzas, R., 2007. Plagiarism Prevention or Detection? The Contribution of Text-Matching Software to Education about Academic Integrity. *Journal of Academic Language and Learning*, Vol 1(1), pp. 91-99.

Library and Information Services, 2010. Cyprus University of Technology in Scopus, http://www.cut.ac.cy/library/english/services/references_en.html#plagiarism.

Loui, M. C., 2002. Seven Ways to Plagiarize: Handling Real Allegations of Research Misconduct. *Science and Engineering Ethics*, 8, pp. 529-539.

Lyon, C., Malcolm, J., and Dickerson, B., 2001. Detecting Short Passages of Similar Text in Large Document Collections. In *Proceedings of Conference on Empirical Methods in Natural Language Processing*, pp. 118-125.

Manber, U., 1994. Finding Similar Files in a Large File System, In *Proceedings of the USENIX Technical Conference*, pp. 1-10.

Monostori1, K., Finkel, R., Zaslavsky, A., Hodasz, G., and Patke, M., 2002. Comparison of Overlap Detection Techniques. In *Proceedings of the 2002 International Conference on Computational Science*, Lecture Notes in Computer Science, vol 2329, pp. 51-60.

Shivakumar, N., and Garcia-Molina, H., 1996. Building a Scalable and Accurate Copy Detection Mechanism. In *Proceedings of the International Conference on Digital Libraries*, pp. 160-168.

Snider, N., and Diab, M., JUNE 2006A. Unsupervised Induction of Modern Standard Arabic Verb Classes. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, pp. 153- 156, June 2006.

Snider, N., and Diab, M., JULY 2006B. Unsupervised Induction of Modern Standard Arabic Verb Classes Using Syntactic Frames and LSA. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pp. 795- 802.

Sorokina, D., Gehrke, J., Warner, S., Ginsparg, P., 2006. Plagiarism Detection in arXiv. In *Proceedings of Sixth International Conference on Data Mining (ICDM)*, pp. 1070-1075.

Wikipedia, 2010. Plagiarism. http://en.wikipedia.org/wiki/Plagiarism.

Witten, I. H., Moffat, A., and Bell, T. C., 1999. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, second edition.

A Structured Vector Space Model for Hidden Attribute Meaning in Adjective-Noun Phrases

Matthias Hartung and **Anette Frank**
Computational Linguistics Department
Heidelberg University
{hartung, frank}@cl.uni-heidelberg.de

Abstract

We present an approach to model hidden attributes in the compositional semantics of adjective-noun phrases in a distributional model. For the representation of *adjective meanings*, we reformulate the pattern-based approach for attribute learning of Almuhareb (2006) in a structured vector space model (VSM). This model is complemented by a structured vector space representing attribute dimensions of *noun meanings*. The combination of these representations along the lines of compositional semantic principles exposes the underlying semantic relations in adjective-noun phrases. We show that our compositional VSM outperforms simple pattern-based approaches by circumventing their inherent sparsity problems.

1 Introduction

In formal semantic theory, the compositional semantics of adjective-noun phrases can be modeled in terms of *selective binding* (Pustejovsky, 1995), i.e. the adjective selects one of possibly several roles or attributes[1] from the semantics of the noun.

(1) a. a blue car
b. COLOR(car)=blue

In this paper, we define a distributional framework that models the compositional process underlying the modification of nouns by adjectives. We focus on property-denoting adjectives as they are valuable for acquiring concept representations for, e.g., ontology learning. An approach for automatic subclassification of property-denoting adjectives is presented in Hartung and Frank (2010). Our goal is to expose, for adjective-noun phrases as in (1a), the attribute in the semantics of the noun that is selected by the adjective, while not being overtly realized on the syntactic level. The semantic information we intend to capture for (1a) is formalized in (1b).

Ideally, this kind of knowledge could be extracted from corpora by searching for patterns that paraphrase (1a), e.g. *the color of the car is blue*. However, linguistic patterns that explicitly relate nouns, adjectives and attributes are very rare.

We avoid these sparsity issues by reducing the triple $r{=}\langle noun, attribute, adjective\rangle$ that encodes the relation illustrated in (1b) to tuples $r'{=}\langle noun, attribute\rangle$ and $r''{=}\langle attribute, adjective\rangle$, as suggested by Turney and Pantel (2010) for similar tasks. Both r' and r'' can be observed much more frequently in text corpora than r. Moreover, this enables us to model adjective and noun meanings as distinct semantic vectors that are built over attributes as dimensions. Based on these semantic representations, we make use of vector composition operations in order to reconstruct r from r' and r''. This, in turn, allows us to infer complete noun-attribute-adjective *triples* from individually acquired noun-attribute and adjective-attribute representations.

The contributions of our work are as follows: (i) We propose a framework for attribute selection based on structured vector space models (VSM), using as meaning dimensions attributes elicited

[1] In the original statement of the theory, adjectives select *qualia roles* that can be considered as collections of attributes.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 430–438, Beijing, August 2010

by adjectives; (ii) we complement this novel representation of adjective meaning with structured vectors for *noun meanings* similarly built on attributes as meaning dimensions; (iii) we propose a composition of these representations that mirrors principles of compositional semantics in mapping adjective-noun phrases to their corresponding ontological representation; (iv) we propose and evaluate several metrics for the selection of meaningful components from vector representations.

2 Related Work

Adjective-noun meaning composition has not been addressed in a distributional framework before (cf. Mitchell and Lapata (2008)). Our approach leans on related work on attribute learning for ontology induction and recent work in distributional semantics.

Attribute learning. Early approaches to attribute learning include Hatzivassiloglou and McKeown (1993), who cluster adjectives that denote values of the same attribute. A weakness of their work is that the type of the attribute cannot be made explicit. More recent attempts to attribute learning from adjectives are Cimiano (2006) and Almuhareb (2006). Cimiano uses attributes as features to arrange sets of concepts in a lattice. His approach to attribute acquisition harnesses adjectives that occur frequently as concept modifiers in corpora. The association of adjectives with their potential attributes is performed by dictionary look-up in WordNet (Fellbaum, 1998). Similarly, Almuhareb (2006) uses adjectives and attributes as (independent) features for the purpose of concept learning. He acquires adjective-attribute pairs using a pattern-based approach.

As a major limitation, these approaches are confined to adjective-attribute pairs. The polysemy of adjectives that can only be resolved in the context of the modified noun is entirely neglected.

From a methodological point of view, our work is similar to Almuhareb's, as we will also build on lexico-syntactic patterns for attribute selection. However, we extend the task to involve nouns and rephrase his approach in a distributional framework based on the composition of structured vector representations.

Distributional semantics. We observe two recent trends in distributional semantics research: (i) The use of VSM tends to shift from measuring unfocused semantic similarity to capturing increasingly fine-grained semantic information by incorporating more linguistic structure. Following Baroni and Lenci (to appear), we refer to such models as *structured vector spaces*. (ii) Distributional methods are no longer confined to word meaning, but are noticeably extended to capture meaning on the *phrase level*. Prominent examples for (i) are Padó and Lapata (2007) and Rothenhäusler and Schütze (2009) who use syntactic dependencies rather than single word co-occurrences as dimensions of semantic spaces. Erk and Padó (2008) extend this idea to the argument structure of verbs, while also accounting for compositional meaning aspects by modelling predication over arguments. Hence, their work is also representative for (ii).

Baroni et al. (2010) use lexico-syntactic patterns to represent concepts in a structured VSM whose dimensions are interpretable as empirical manifestations of properties. We rely on similar techniques for the acquisition of structured vectors, whereas our work focusses on exposing the hidden meaning dimensions involved in compositional processes underlying concept modification.

The commonly adopted method for modelling compositionality in VSM is vector composition (Mitchell and Lapata, 2008; Widdows, 2008). Showing the benefits of vector composition for language modelling, Mitchell and Lapata (2009) emphasize its potential to become a standard method in NLP.

The approach pursued in this paper builds on both lines of research sketched in (i) and (ii) in that we model a specific meaning layer in the semantics of adjectives and nouns in a structured VSM. Vector composition is used to expose their hidden meaning dimensions on the phrase level.

3 Structured Vector Representations for Adjective-Noun Meaning

3.1 Motivation

Contrary to prior work, we model attribute selection as involving *triples* of nouns, attributes and

	COLOR	DIRECTION	DURATION	SHAPE	SIZE	SMELL	SPEED	TASTE	TEMPERATURE	WEIGHT
v_e	1	1	0	1	45	0	4	0	0	21
v_b	14	38	2	20	26	0	45	0	0	20
$v_e \times v_b$	14	38	0	20	**1170**	0	180	0	0	420
$v_e + v_b$	15	39	2	21	**71**	0	49	0	0	41

Figure 1: Vectors for *enormous* (v_e) and *ball* (v_b)

adjectives, as in (2). The triple r can be broken down into tuples $r' = \langle noun, attribute \rangle$ and $r'' = \langle attribute, adjective \rangle$. Previous learning approaches focussed on r' (Cimiano, 2006) or r'' (Almuhareb, 2006) only.

(2) a. a $\text{blue}_{\text{value}}$ $\text{car}_{\text{concept}}$
b. ATTR(concept) = value

In semantic composition of adjective-noun compounds, the adjective (e.g. *blue*) contributes a value for an attribute (here: COLOR) that characterizes the concept evoked by the noun (e.g. *car*). Thus, the attribute in (2) constitutes a 'hidden variable' that is not overtly expressed in (2a), but constitutes the central axis that relates r' and r''.

Structured vectors built on extraction patterns. We model the semantics of adjectives and nouns in a structured VSM that conveys the hidden relationship in (2). The dimensions of the model are defined by attributes, such as COLOR, SIZE or SPEED, while the vector components are determined on the basis of carefully selected acquisition patterns that are tailored to capturing the particular semantic information of interest for r' and r''. In this respect, lexico-syntactic patterns serve a similar purpose as dependency relations in Padó and Lapata (2007) or Rothenhäusler and Schütze (2009). The upper part of Fig. 1 displays examples of vectors we build for adjectives and nouns.

Composing vectors along hidden dimensions. The fine granularity of lexico-syntactic patterns that capture the triple r comes at the cost of their sparsity when applied to corpus data. Therefore, we construct separate vector representations for r' and r''. Eventually, these representations are joined by vector composition to reconstruct the triple r. Apart from avoiding sparsity issues, this compositional approach has several prospects from a linguistic perspective as well.

Ambiguity and disambiguation. Building vectors with attributes as meaning dimensions enables us to model (i) ambiguity of adjectives with regard to the attributes they select, and (ii) the disambiguation capacity of adjective and noun vectors when considered jointly. Consider, for example, the phrase *enormous ball* that is ambiguous for two reasons: *enormous* may select a set of possible attributes (SIZE or WEIGHT, among others), while *ball* elicits several attributes in accordance with its different word senses[2]. As seen in Fig. 1, these ambiguities are nicely captured by the separate vector representations for the adjective and the noun (upper part); by composing these representations, the ambiguity is resolved (lower part).

3.2 Building a VSM for Adjective-Noun Meaning

In this section, we introduce the methods we apply in order to (i) acquire vector representations for adjectives and nouns, (ii) select appropriate attributes from them, and (iii) compose them.

3.2.1 Attribute Acquisition Patterns

We use the following patterns[3] for the acquisition of vectors capturing the tuple $r'' = \langle attribute, adjective \rangle$. Even though some of these patterns (`A1` and `A4`) match triples of nouns, attributes and adjectives, we only use them for the extraction of binary tuples (underlined), thus abstracting from the modified noun.

(A1) `ATTR of DT? NN is|was JJ`
(A2) `DT? RB? JJ ATTR`
(A3) `DT? JJ or JJ ATTR`
(A4) `DT? NN's ATTR is|was JJ`
(A5) `is|was|are|were JJ in|of ATTR`

To acquire noun vectors capturing the tuple $r' = \langle noun, attribute \rangle$, we rely on the following patterns. Again, we only extract pairs, as indicated by the underlined elements.

(N1) `NN with|without DT? RB? JJ? ATTR`
(N2) `DT ATTR of DT? RB? JJ? NN`
(N3) `DT NN's RB? JJ? ATTR`
(N4) `NN has|had a|an RB? JJ? ATTR`

[2]WordNet senses for the noun *ball* include, among others: 1. *round object [...] in games*; 2. *solid projectile*, 3. *object with a spherical shape*, 4. *people [at a] dance*.

[3]Some of these patterns are taken from Almuhareb (2006) and Sowa (2000). The descriptions rely on the Penn Tagset (Marcus et al., 1999). `?` marks optional elements.

3.2.2 Target Filtering

Some of the adjectives extracted by `A1-A5` are not property-denoting and thus represent noise. This affects in particular pattern `A2`, which extracts adjectives like *former* or *more*, or relational ones such as *economic* or *geographic*.

This problem may be addressed in different ways: By *target filtering*, extractions can be checked against a predicative pattern `P1` that is supposed to apply to property-denoting adjectives only. Vectors that fail this test are suppressed.

(P1) `DT NN is|was JJ`

Alternatively, extractions obtained from low-confidence patterns can be awarded reduced weights by means of a *pattern value function* (defined in 3.3; cf. Pantel and Pennacchiotti (2006)).

3.2.3 Attribute Selection

We intend to use the acquired vectors in order to detect attributes that are implicit in adjective-noun meaning. Therefore, we need a method that selects appropriate attributes from each vector. While, in general, this task consists in distinguishing semantically meaningful dimensions from noise, the requirements are different depending on whether attributes are to be selected from adjective or noun vectors. This is illustrated in Fig. 1, a typical configuration, with one vector representing a typical property-denoting adjective that exhibits relatively strong peaks on one or more dimensions, whereas noun vectors show a tendency for broad and flat distributions over their dimensions. This suggests using a strict selection function (choosing few very prominent dimensions) for adjectives and a less restrictive one (licensing the inclusion of more dimensions of lower relative prominence) for nouns. Moreover, we are interested in finding a selection function that relies on as few free parameters as possible in order to avoid frequency or dimensionality effects.

MPC Selection (MPC). An obvious method for attribute selection is to choose the most prominent component from any vector (i.e., the highest absolute value). If a vector exhibits several peaks, all other components are rejected, their relative importance notwithstanding. MPC obviously fails to capture polysemy of targets, which affects adjectives such as *hot*, in particular.

Threshold Selection (TSel). TSel recasts the approach of Almuhareb (2006), in selecting all dimensions as attributes whose components exceed a frequency threshold. This avoids the drawback of MPC, but introduces a parameter that needs to be optimized. Also, it is difficult to apply absolute thresholds to composed vectors, as the range of their components is subject to great variation, and it is unclear whether the method will scale with increased dimensionality.

Entropy Selection (ESel). In information theory, entropy measures the average uncertainty in a probability distribution (Manning and Schütze, 1999). We define the entropy $H(v)$ of a vector $v{=}\langle v_1,\ldots,v_n\rangle$ over its components as $H(v) = -\sum_{i=1}^{n} P(v_i)\log P(v_i)$, where $P(v_i) = v_i / \sum_{i=1}^{n} v_i$.

We use $H(v)$ to assess the impact of singular vector components on the overall entropy of the vector: We expect entropy to detect components that contribute noise, as opposed to those that contribute important information.

We define an algorithm for entropy-based attribute selection that returns a list of informative dimensions. The algorithm successively suppresses (combinations of) vector components one by one. Given that a gain of entropy is equivalent to a loss of information and vice versa, we assume that every combination of components that leads to an increase in entropy when being suppressed is actually responsible for a substantial amount of information. The algorithm includes a back-off to MPC for the special case that a vector contains a single peak (i.e., $H(v) = 0$), so that, in principle, it should be applicable to vectors of any kind. Vectors with very broad distributions over their dimensions, however, pose a problem to this method. For *ball* in Fig. 1, for instance, the method does not select any dimension.

Median Selection (MSel). As a further method we rely on the median m that can be informally defined as the value that separates the upper from the lower half of a distribution (Krengel, 2003). It is less restrictive than MPC and TSel and overcomes the particular drawback of ESel. Using this measure, we choose all dimensions whose components exceed m. Thus, for the vector representing

Pattern Label	# Hits (Web)	# Hits (ukWaC)
A1	2249	815
A2	36282	72737
A3	3370	1436
A4	–	7672
A5	–	3768
N1	–	682
N2	–	5073
N3	–	953
N4	–	56

Table 1: Number of pattern hits on the Web (Almuhareb, 2006) and on ukWaC

ball, WEIGHT, DIRECTION, SHAPE, SPEED and SIZE are selected.

3.2.4 Vector Composition

We use vector composition as a hinge to combine adjective and noun vectors in order to reconstruct the triple $r = \langle noun, attribute, adjective \rangle$. Mitchell and Lapata (2008) distinguish two major classes of vector composition operations, namely multiplicative and additive operations, that can be extended in various ways. We use their standard definitions (denoted $\times$ and $+$, henceforth). For our task, we expect $\times$ to perform best as it comes closest to the linguistic function of *intersective* adjectives, i.e. to select dimensions that are prominent both for the adjective and the noun, whereas $+$ basically blurs the vector components, as can be seen in the lower part of Fig. 1.

3.3 Model Parameters

We follow Padó and Lapata (2007) in defining a semantic space as a matrix $M = B \times T$ relating a set of target elements T to a set of basis elements B. Further parameters and their instantiations we use in our model are described below. We use p to denote an individual lexico-syntactic pattern.

The **basis elements** of our VSM are nouns denoting attributes. For comparison, we use the attributes selected by Almuhareb (2006): COLOR, DIRECTION, DURATION, SHAPE, SIZE, SMELL, SPEED, TASTE, TEMPERATURE, WEIGHT.

The **context selection function** *cont(t)* determines the set of patterns that contribute to the representation of each target word $t \in T$. These are the patterns `A1`-`A5` and `N1`-`N4` (cf. Section 3.2.1).

The **target elements** represented in the vector space comprise all adjectives T_A that match the patterns `A1` to `A5` in the corpus, provided they exceed a frequency threshold n. During development, n was set to 5 in order to filter noise.

As for the target nouns T_N, we rely on a representative dataset compiled by Almuhareb (2006). It contains 402 nouns that are balanced with regard to semantic class (according to the WordNet supersenses), ambiguity and frequency.

As **association measure** that captures the strength of the association between the elements of B and T, we use raw frequency counts[4] as obtained from the PoS-tagged and lemmatized version of the ukWaC corpus (Baroni et al., 2009). Table 1 gives an overview of the number of hits returned by these patterns.

The **basis mapping function** μ creates the dimensions of the semantic space by mapping each extraction of a pattern p to the attribute it contains.

The **pattern value function** enables us to subdivide dimensions along particular patterns. We experimented with two instantiations: pv_{const} considers, for each dimension, all patterns, while weighting them equally. $pv_f(p)$ awards the extractions of pattern p with weight 1, while setting the weights for all patterns different from p to 0.

4 Experiments

We evaluate the performance of the structured VSM on the task of inferring attributes from adjective-noun phrases in three experiments: In Exp1 and Exp2, we evaluate vector representations capturing r' and r'' independently of one another. Exp3 investigates the selection of hidden attributes from vector representations constructed by composition of adjective and noun vectors.

We compare all results against different *gold standards*. In Exp1, we follow Almuhareb (2006), evaluating against WordNet 3.0. For Exp2 and Exp3, we establish gold standards manually: For Exp2, we construct a test set of nouns annotated with their corresponding attributes. For Exp3, we manually annotate adjective-noun phrases with the attributes appropriate for the whole phrase. All experiments are evaluated in terms of precision, recall and F_1 score.

[4]We experimented with the conditional probability ratio proposed by Mitchell and Lapata (2009). As it performed worse on our data, we did not consider it any further.

4.1 Exp1: Attribute Selection for Adjectives

The first experiment evaluates the performance of structured vector representations on attribute selection for adjectives. We compare this model against a re-implementation of Almuhareb (2006).

Experimental settings and gold standard. To reconstruct Almuhareb's approach, we ran his patterns A1-A3 on the ukWaC corpus. Table 1 shows the number of hits when applied to the Web (Almuhareb, 2006) vs. ukWaC. A1 and A3 yield less extractions on ukWaC as compared to the Web.[5] We introduced two additional patterns, A4 and A5, that contribute about 10,000 additional hits. We adopted Almuhareb's manually chosen thresholds for attribute selection for A1-A3; for A4, A5 and a combination of all patterns, we manually selected optimal thresholds.

We experiment with pv_{const} and all variants of $pv_f(p)$ for pattern weighting (see sect. 3.3). For attribute selection, we compare TSel (as used by Almuhareb), ESel and MSel.

The gold standard consists of all adjectives that are linked to at least one of the ten attributes we consider by WordNet's `attribute` relation (1063 adjectives in total).

Evaluation results. Results for Exp1 are displayed in Table 2. The settings of pv are given in the rows, the attribute selection methods (in combination with target filtering[6]) in the columns.

The results for our re-implementation of Almuhareb's individual patterns are comparable to his original figures[7], except for A3 that seems to suffer from quantitative differences of the underlying data. Combining all patterns leads to an improvement in precision over (our reconstruction of) Almuhareb's best individual pattern when TSel and target filtering are used in combination. MPC and MSel perform worse (not reported here). As for target filtering, A1 and A3 work best.

Both TSel and ESel benefit from the combination with the target filter, where the largest improvement (and the best overall result) is observable for ESel on pattern A1 only. This is the pattern that performs worst in Almuhareb's original setting. From this, we conclude that both ESel and target filtering are valuable extensions to pattern-based structured vector spaces if precision is in focus. This also underlines a finding of Rothenhäusler and Schütze (2009) that VSMs intended to convey specific semantic information rather than mere similarity benefit primarily from a linguistically adequate choice of contexts.

Similar to Almuhareb, recall is problematic. Even though ESel leads to slight improvements, the scores are far from satisfying. With Almuhareb, we note that this is mainly due to a high number of extremely fine-grained adjectives in WordNet that are rare in corpora.[8]

[5] The difference for A2 is an artifact of Almuhareb's extraction methodology.

[6] Regarding target filtering, we only report the best filter pattern for each configuration.

[7] P(A1)=0.176, P(A2)=0.218, P(A3)=0.504

	MPC			ESel			MSel		
	P	R	F	P	R	F	P	R	F
pv_f(N1)	0.22	0.06	0.10	**0.29**	0.04	0.07	0.22	**0.09**	**0.13**
pv_f(N2)	**0.29**	0.18	0.23	0.20	0.06	0.09	0.28	**0.39**	**0.33**
pv_f(N3)	**0.34**	0.05	0.09	0.20	0.02	0.04	0.25	**0.08**	**0.12**
pv_f(N4)	0.25	0.02	0.04	**0.29**	0.02	0.03	0.26	0.02	**0.05**
pv_{const}	**0.29**	0.18	0.22	0.20	0.06	0.09	0.28	**0.43**	**0.34**

Table 3: Evaluation results for Experiment 2

4.2 Exp2: Attribute Selection for Nouns

Exp2 evaluates the performance of attribute selection from noun vectors tailored to the tuple r''.

Construction of the gold standard. For evaluation, we created a gold standard by manually annotating a set of nouns with attributes. This gold standard builds on a random sample extracted from T_N (cf. section 3.3). Running N1-N4 on ukWaC returned semantic vectors for 216 concepts. From these, we randomly sampled 100 concepts that were manually annotated by three human annotators.

The annotators were provided a matrix consisting of the nouns and the set of ten attributes for each noun. Their task was to remove all inappropriate attributes. They were free to decide how many attributes to accept for each noun. In order to deal with word sense ambiguity, the annotators were instructed to consider all senses of a noun and to retain every attribute that was acceptable for at least one sense.

Inter-annotator agreement amounts to κ= 0.69 (Fleiss, 1971). Cases of disagreement were adjudicated by majority-voting. The gold standard

	Almuhareb (reconstr.)				VSM (TSel + Target Filter)					VSM (ESel)			VSM (ESel + Target Filter)			
	P	R	F	Thr	P	R	F	Patt	Thr	P	R	F	P	R	F	Patt
$pv_f(\mathtt{A1}) = 1$	0.183	0.005	0.009	5	0.300	0.004	0.007	A3	5	0.231	**0.045**	**0.076**	<u>**0.519**</u>	0.035	0.065	A3
$pv_f(\mathtt{A2}) = 1$	0.207	0.039	0.067	50	**0.300**	0.033	0.059	A1	50	0.084	**0.136**	**0.104**	0.240	0.049	0.081	A3
$pv_f(\mathtt{A3}) = 1$	0.382	0.020	0.039	5	**0.403**	0.014	0.028	A1	5	0.192	**0.059**	**0.090**	0.375	0.027	0.050	A1
$pv_f(\mathtt{A4}) = 1$					**0.301**	0.020	0.036	A3	10	0.135	**0.055**	**0.078**	0.272	0.020	0.038	A1
$pv_f(\mathtt{A5}) = 1$					0.295	0.008	0.016	A3	24	0.105	**0.056**	**0.073**	**0.315**	0.024	0.045	A3
pv_{const}					**0.420**	0.024	0.046	A1	183	0.076	**0.152**	**0.102**	0.225	0.054	0.087	A3

Table 2: Evaluation results for Experiment 1

contains 424 attributes for 100 nouns.

Evaluation results. Results for Exp2 are given in Table 3. Performance is lower in comparison to Exp1. We hypothesize that the tuple r'' might not be fully captured by overt linguistic patterns. This needs further investigation in future research.

Against this background, MPC is relatively precise, but poor in terms of recall. ESel, being designed to select more than one prominent dimension, counterintuitively fails to increase recall, suffering from the fact that many noun vectors show a rather flat distribution without any strong peak. MSel turns out to be most suitable for this task: Its precision is comparable to MPC (with N3 as an outlier), while recall is considerably higher. Overall, these results indicate that attribute selection for adjectives and nouns, though similar, should be viewed as distinct tasks that require different attribute selection methods.

4.3 Exp3: Attribute Selection for Adjective-Noun Phrases

In this experiment, we compose noun and adjective vectors in order to yield a new combined representation. We investigate whether the semantic information encoded by the components of this new vector is sufficiently precise to disambiguate the attribute dimensions of the original representations (see section 3.1) and, thus, to infer hidden attributes from adjective-noun phrases (see (2)) as advocated by Pustejovsky (1995).

Construction of the gold standard. For evaluation, we created a manually annotated test set of adjective-noun phrases. We selected a subset of property-denoting adjectives that are appropriate modifiers for the nouns from T_N using the predicative pattern P1 (see sect. 3) on ukWaC. This yielded 2085 adjective types that were further reduced to 386 by frequency filtering ($n = 5$). We sampled our test set from all pairs in the cartesian product of the 386 adjectives and 216 nouns (cf. Exp2) that occurred at least 5 times in a subsection of ukWaC. To ensure a sufficient number of ambiguous adjectives in the test set, sampling proceeded in two steps: First, we sampled four nouns each for a manual selection of 15 adjectives of all ambiguity levels in WordNet. This leads to 60 adjective-noun pairs. Second, another 40 pairs were sampled fully automatically.

The test set was manually annotated by the same annotators as in Exp2. They were asked to remove all attributes that were not appropriate for a given adjective-noun pair, either because it is not appropriate for the noun or because it is not selected by the adjective. Further instructions were as in Exp2, in particular regarding ambiguity.

The overall agreement is κ=0.67. After adjudication by majority voting, the resulting gold standard contains 86 attributes for 76 pairs. 24 pairs could not be assigned any attribute, either because the adjective did not denote a property, as in *private investment*, or the most appropriate attribute was not offered, as in *blue day* or *new house*.

We evaluate the vector composition methods discussed in section 3.2.4. Individual vectors for the adjectives and nouns from the test pairs were constructed using all patterns A1-A5 and N1-N4. For attribute selection, we tested MPC, ESel and MSel. The results are compared against three baselines: BL-P implements a purely pattern-based method, i.e. running the patterns that extract the triple r (A1, A4, N1, N3 and N4, with JJ and NN instantiated accordingly) on the pairs from the test set. BL-N and BL-Adj are back-offs for vector composition, taking the respective noun or adjective vector, as investigated in Exp1 and Exp2, as surrogates for a composed vector.

[8] For instance: *bluish-lilac*, *chartreuse* or *pink-lavender* as values of the attribute COLOR.

	MPC			ESel			MSel		
	P	R	F	P	R	F	P	R	F
$\times$	0.60	0.58	**0.59**	**0.63**	0.46	0.54	0.27	0.72	0.39
$+$	0.43	0.55	0.48	0.42	0.51	0.46	0.18	**0.91**	0.30
BL-Adj	0.44	0.60	0.50	0.51	0.63	0.57	0.23	0.83	0.36
BL-N	0.27	0.35	0.31	0.37	0.29	0.32	0.17	0.73	0.27
BL-P	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 4: Evaluation results for Experiment 3

Evaluation results. Results are given in Table 4. Attribute selection based on the composition of adjective and noun vectors yields a considerable improvement of both precision and recall as compared to the individual results obtained in Exp1 and Exp2. Comparing the results of Exp3 against the baselines reveals two important aspects of our work. First, the complete failure of BL-P[9] underlines the attractiveness of our method to build structured vector representations from patterns of reduced complexity. Second, vector composition is suitable for selecting hidden attributes from adjective-noun phrases that are jointly encoded by adjective and noun vectors: Both composition methods we tested outperform BL-N.

However, the choice of the composition method matters: $\times$ performs best with a maximum precision of 0.63. This confirms our expectation that vector multiplication is a good approximation for attribute selection in adjective-noun semantics. Being outperformed by BL-Adj in most categories, $+$ is less suited for this task.

All selection methods outperform BL-Adj in precision. Comparing MPC and ESel, ESel achieves better precision when combined with the $\times$-operator, while doing worse for recall. The robust performance of MPC is not surprising as the test set contains only ten adjective-noun pairs that are still ambiguous with regard to the attributes they elicit. The stronger performance of the entropy-based method with the $\times$-operator is mainly due to its accuracy on detecting false positives, in that it is able to return ”empty” selections. In terms of precision, MSel did worse in general, while recall is decent. This underlines that vector composition generally promotes meaningful components, but MSel is too inaccurate to select them.

Given the performance of the baselines and the noun vectors in Exp2, we consider this a very promising result for our approach to attribute selection from structured vector representations. The results also corroborate the insufficiency of previous approaches to attribute learning from adjectives alone.

[9]The patterns used yield no hits for the test pairs at all.

5 Conclusions and Outlook

We proposed a structured VSM as a framework for inferring hidden attributes from the compositional semantics of adjective-noun phrases.

By reconstructing Almuhareb (2006), we showed that structured vector representations of adjective meaning consistently outperform simple pattern-based learning, up to 13 pp. in precision. A combination of target filtering and pattern weighting turned out to be effective here, by selecting particulary meaningful lexico-syntactic contexts and filtering adjectives that are not property-denoting. Further studies need to investigate this phenomenon and its most appropriate formulation in a vector space framework.

Moreover, the VSM offers a natural representation for sense ambiguity of adjectives. Comparing attribute selection methods on adjective and noun vectors shows that they are sensitive to the distributional structure of the vectors, and need to be chosen with care. Future work will investigate these selection methods in high-dimensional vectors spaces, by using larger sets of attributes.

Exp3 shows that the composition of pattern-based adjective and noun vectors robustly reflects aspects of meaning composition in adjective-noun phrases, with attributes as a hidden dimension. It also suggests that composition is effective in disambiguation of adjective and noun meanings. This hypothesis needs to be substantiated in further experiments.

Finally, we showed that composition of vectors representing complementary meaning aspects can be beneficial to overcome sparsity effects. However, our compositional approach meets its limits if the patterns capturing adjective and noun meaning in isolation are too sparse to acquire sufficiently populated vector components from corpora. For future work, we envisage using vector similarity to acquire structured vectors for infrequent targets from semantic spaces that convey less linguistic structure to address these remaining sparsity issues.

References

Almuhareb, Abdulrahman. 2006. *Attributes in Lexical Acquisition*. Ph.D. Dissertation, Department of Computer Science, University of Essex.

Baroni, Marco and Alessandro Lenci. to appear. Distributional Memory. A General Framework for Corpus-based Semantics. *Computational Linguistics*.

Baroni, Marco, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: A collection of very large linguistically processed web-crawled corpora. *Journal of Language Resources and Evaluation*, 43(3):209–226.

Baroni, Marco, Brian Murphy, Eduard Barbu, and Massimo Poesio. 2010. Strudel. A Corpus-based Semantic Model of Based on Properties and Types. *Cognitive Science*, 34:222–254.

Cimiano, Philipp. 2006. *Ontology Learning and Population from Text. Algorithms, Evaluation and Applications*. Springer.

Erk, Katrin and Sebastian Padó. 2008. A Structured Vector Space Model for Word Meaning in Context. In *Proceedings of EMNLP*, Honolulu, HI.

Fellbaum, Christiane, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, Mass.

Fleiss, Joseph L. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Hartung, Matthias and Anette Frank. 2010. A Semi-supervised Type-based Classification of Adjectives. Distinguishing Properties and Relations. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, Valletta, Malta, May.

Hatzivassiloglou, Vasileios and Kathleen McKeown. 1993. Towards the Automatic Identification of Adjectival Scales. Clustering Adjectives According to Meaning. In *Proceedings of the 31st Annual Meeting of the Association of Computational Linguistics*, pages 172–182.

Krengel, Ulrich. 2003. *Wahrscheinlichkeitstheorie und Statistik*. Vieweg, Wiesbaden.

Manning, Christopher D. and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.

Marcus, Mitchell P., Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. Treebank-3, ldc99t42. CD-ROM. Philadelphia, Penn.: Linguistic Data Consortium.

Mitchell, Jeff and Mirella Lapata. 2008. Vector-based Models of Semantic Composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June.

Mitchell, Jeff and Mirella Lapata. 2009. Language Models Based on Semantic Composition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, Singapore, August 2009, pages 430–439, Singapore, August.

Padó, Sebastian and Mirella Lapata. 2007. Dependency-based Construction of Semantic Space Models. *Computational Linguistics*, 33:161–199.

Pantel, Patrick and Marco Pennacchiotti. 2006. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, 17–21 July 2006, pages 113–120.

Pustejovsky, James. 1995. *The Generative Lexicon*. MIT Press, Cambridge, Mass.

Rothenhäusler, Klaus and Hinrich Schütze. 2009. Unsupervised Classification with Dependency Based Word Spaces. In *Proceedings of the EACL Workshop on Geometrical Models of Natural Language Semantics (GEMS)*, pages 17–24, Athens, Greece, March.

Sowa, John F. 2000. *Knowledge Representation. Logical, Philosophical, and Computational Foundations*. Brooks Cole.

Turney, Peter D. and Patrick Pantel. 2010. From Frequency to Meaning. Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Widdows, Dominic. 2008. Semantic Vector Products. Some Initial Investigations. In *Proceedings of the 2nd Conference on Quantum Interaction*, Oxford, UK, March.

Hierarchical Phrase-based Machine Translation with Word-based Reordering Model

Katsuhiko Hayashi*, Hajime Tsukada**
Katsuhito Sudoh, Kevin Duh**, Seiichi Yamamoto***
*Doshisha University
katsuhiko-h@is.naist.jp, seyamamo@mail.doshisha.ac.jp
**NTT Communication Science Laboratories
tsukada, sudoh, kevinduh@cslab.kecl.ntt.co.jp

Abstract

Hierarchical phrase-based machine translation can capture global reordering with synchronous context-free grammar, but has little ability to evaluate the correctness of word orderings during decoding. We propose a method to integrate word-based reordering model into hierarchical phrase-based machine translation to overcome this weakness. Our approach extends the synchronous context-free grammar rules of hierarchical phrase-based model to include reordered source strings, allowing efficient calculation of reordering model scores during decoding. Our experimental results on Japanese-to-English basic travel expression corpus showed that the BLEU scores obtained by our proposed system were better than those obtained by a standard hierarchical phrase-based machine translation system.

1 Introduction

Hierarchical phrase-based machine translation (Chiang, 2007; Watanabe et al., 2006) is one of the promising statistical machine translation approaches (Brown et al., 1993). Its model is formulated by a synchronous context-free grammar (SCFG) which captures the syntactic information between source and target languages. Although the model captures global reordering by SCFG, it does not explicitly introduce reordering model to constrain word order. In contrast, lexicalized reordering models (Tillman, 2004; Koehn et al., 2005; Nagata et al., 2006) are extensively used for phrase-based translation. These lexicalized reordering models cannot be directly applied to hierarchical phrased-based translation since the hierarchical phrase representation uses nonterminal symbols.

To handle global reordering in phrase-based translation, various preprocessing approaches have been proposed, where the source sentence is reordered to target language order beforehand (Xia and McCord, 2004; Collins et al., 2005; Li et al., 2007; Tromble and Eisner, 2009). However, preprocessing approaches cannot utilize other information in the translation model and target language model, which has been proven helpful in decoding.

This paper proposes a method that incorporates word-based reordering model into hierarchical phrase-based translation to constrain word order. In this paper, we adopt the reordering model originally proposed by Tromble and Eisner (2009) for the preprocessing approach in phrase-based translation. To integrate the word-based reordering model, we added a reordered source string into the right-hand-side of SCFG's rules. By this extension, our system can generate the reordered source sentence as well as target sentence and is able to efficiently calculate the score of the reordering model. Our method utilizes the translation model and target language model as well as the reordering model during decoding. This is an advantage of our method over the preprocessing approach.

The remainder of this paper is organized as follows. Section 2 describes the concept of our approach. Section 3 briefly reviews our proposed method on hierarchical phrase-based ma-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 439–446,
Beijing, August 2010

Standard SCFG	$X \rightarrow <$ $X1$ wa jinsei no $X2$ da , $X1$ is $X2$ of life$>$
SCFG (move-to-front)	$X \rightarrow <$ $X1$ wa jinsei no $X2$ da , wa $X1$ da $X2$ no jinsei , $X1$ is $X2$ of life$>$
SCFG (attach)	$X \rightarrow <$ $X1$ wa jinsei no $X2$ da , $X1$ wa da $X2$ no jinsei , $X1$ is $X2$ of life$>$

Table 1: A Japanese-to-English example of various SCFG's rule representations. Japanese words are romanized. Our proposed representation of rules has reordered source string to generate reordered source sentence $S^{'}$ as well as target sentence T. The "move-to-front" means Tromble and Eisner (2009) 's algorithm and the "attach" means Al-Onaizan and Papineni (2006) 's algorithm.

chine translation model. We experimentally compare our proposed system to a standard hierarchical phrase-based system on Japanese-to-English translation task in Section 4. Then we discuss on related work in Section 5 and conclude this paper in Section 6.

2 The Concept of Our Approach

The preprocessing approach (Xia and McCord, 2004; Collins et al., 2005; Li et al., 2007; Tromble and Eisner, 2009) splits translation procedure into two stages:

$$S \rightarrow S^{'} \rightarrow T \quad (1)$$

where S is a source sentence, $S^{'}$ is a reordered source sentence with respect to the word order of target sentence T. Preprocessing approach has the very deterministic and hard decision in reordering. To overcome the problem, Li et al. (2007) proposed k-best appoach. However, even with a k-best approach, it is difficult to generate good hypotheses $S^{'}$ by using only a reordering model.

In this paper, we directly integrated the reordering model into the decoder in order to use the reordering model together with other information in the hierarchical phrase-based translation model and target language model. Our approach is expressed as the following equation.

$$S \rightarrow (S^{'}, T). \quad (2)$$

Our proposed method generates the reordered source sentence $S^{'}$ by SCFG and evaluates the correctness of the reorderings using a word-based reordering model of S' which will be introduced in section 3.4.

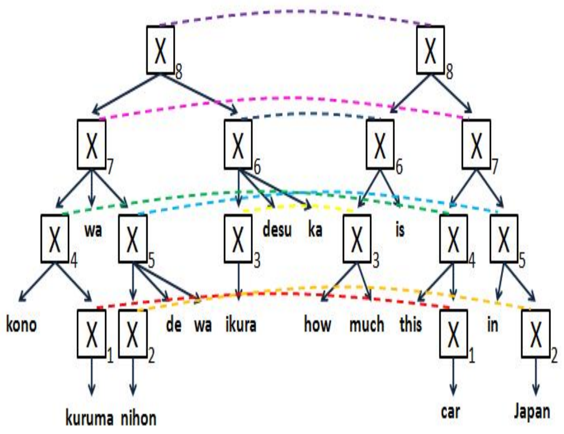


Figure 1: A derivation tree for Japanse-to-English translation.

3 Hierarchical Phrase-based Model Extension

3.1 Hierarchical Phrase-based Model

Hierarchical phrase-based model (Chiang, 2007) induces rules of the form

$$X \rightarrow < \gamma, \alpha, \sim, w > \quad (3)$$

where X is a non-terminal symbol, γ is a sequence string of non-terminals and source terminals, α is a sequence string of non-terminals and target terminals. $\sim$ is a one-to-one correspondence for the non-terminals appeared in γ and α.

Given a source sentence S, the translation task under this model can be expressed as

$$\hat{T} = T\left(\underset{D:S(D)=S}{\operatorname{argmax}}\ w(D)\right) \quad (4)$$

where D is a derivation and $w(D)$ is a score of the derivation. Decoder seeks a target sentence

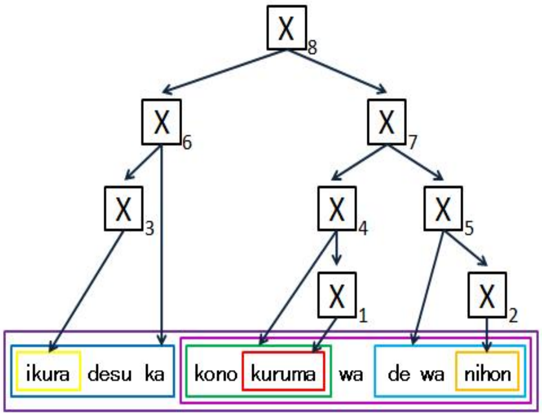


Figure 2: Reordered source sentence generated by our proposed system.

$T(D)$ which has the highest score $w(D)$. $S(D)$ is a source sentence under a derivation D. Figure 1 shows the example of Japanese-to-English translation by hierarchical phrase-based machine translation model.

3.2 Rule Extension

To generate reordered source sentence $S^{'}$ as well as target sentence T, we extend hierarchical phrase rule expressed in Equation 3 to

$$X \rightarrow < \gamma, \gamma^{'}, \alpha, \sim, w > \quad (5)$$

where $\gamma^{'}$ is a sequence string of non-terminals and source terminals, which is reordered γ with respect to the word order of target string α. The reason why we add $\gamma^{'}$ to rules is to efficiently calculate the reordering model scores. If each rule does not have $\gamma^{'}$, the decoder need to keep word alignments because we cannot know word order of $S^{'}$ without them. The calculation of reordering model scores using word alignments is very wasteful when decoding.

The translation task under our model extends Equation 4 to the following equation:

$$\hat{T} = (\hat{S}^{'}, \hat{T}) = (S^{'}, T)\left(\underset{D:S(D)=S}{\operatorname{argmax}}\ w(D)\right). \quad (6)$$

Our system generates the reordered source sentence $S^{'}$ as well as target sentence T. Figure 2 shows the generated reordered source sentence $S^{'}$ when translating the example of Figure 1. Note that the structure of $S^{'}$ is the same as that of target sentence T. The decoder generates both Figure 2 and the right hand side of Figure 1, allowing us to score both global and local word reorderings.

To add $\gamma^{'}$ to rules, we permuted γ into $\gamma^{'}$ after rule extraction based on Grow-diag-final (Koehn et al., 2005) alignment by GIZA++ (Och and Ney, 2003). To do this permutation on rules, we applied two methods. One is the same algorithm as Tromble and Eisner (2009), which reorders aligned source terminals and nonterminals in the same order as that of target side and moves unaligned source terminals to the front of aligned terminals or nonterminals (move-to-front). The other is the same algorithm as AI-Onaizan and Papineni (2006), which differs from Tromble and Eisner's approach in attaching unaligned source terminals to the closest prealigned source terminals or nonterminals (attach). This extension of adding $\gamma^{'}$ does not increase the number of rules.

Table 1 shows a Japanese-to-English example of the representation of rules for our proposed system. Japanese words are romanized. Suppose that source-side string is (X1 wa jinsei no X2 da) and target-side string is (X1 is X2 of life) and their word alignments are a=((jinsei , life) , (no , of) , (da , is)). Source-side aligned words and nonterminal symbols are sorted into the same order of target string. Source-side unaligned word (wa) is moved to the front or right of the prealigned symbol (X1).

Uni-gram Features	Bi-gram Features
s_r, $s\text{-}pos_r$	s_r, $s\text{-}pos_r$, s_l, $s\text{-}pos_l$
s_r	$s\text{-}pos_r$, s_l, $s\text{-}pos_l$
$s\text{-}pos_r$	s_r, s_l, $s\text{-}pos_l$
s_l, $s\text{-}pos_l$	s_r, $s\text{-}pos_r$, $s\text{-}pos_l$
s_l	s_r, $s\text{-}pos_r$, s_l
$s\text{-}pos_l$	s_r, s_l
	$s\text{-}pos_r$, $s\text{-}pos_l$

Table 2: Features used by Word-based Reordering Model. pos means part-of-speech tag.

Surrounding Word Pos Features
$s\text{-}pos_r$, $s\text{-}pos_r+1$, $s\text{-}pos_l-1$, $s\text{-}pos_l$
$s\text{-}pos_r-1$, $s\text{-}pos_r$, $s\text{-}pos_l-1$, $s\text{-}pos_l$
$s\text{-}pos_r$, $s\text{-}pos_r+1$, $s\text{-}pos_l$, $s\text{-}pos_l+1$
$s\text{-}pos_r-1$, $s\text{-}pos_r$, $s\text{-}pos_l$, $s\text{-}pos_l+1$

Table 3: The Example of Context Features

3.3 Word-based Reordering Model

We utilize the following $score(S^{'})$ as a feature for the word-based reordering model. This is incorporated into the log-linear model (Och and Ney, 2002) of statistical machine translation.

$$score(S^{'}) = \sum_{i,j:1\leq i<j\leq n} B[s_i^{'}, s_j^{'}] \tag{7}$$

$$B[s_l^{'}, s_r^{'}] = \theta \cdot \phi(s_l^{'}, s_r^{'}) \tag{8}$$

where n is the length of reordered source sentence $S^{'}$ $(= (s_1^{'} \dots s_n^{'}))$, θ is a weight vector and ϕ is a vector of features. This reordering model, which is originally proposed by Tromble and Eisner (2009), can assign a score to any possible permutation of source sentences. Intuitively $B[s_l^{'}, s_r^{'}]$ represents the score of ordering $s_l^{'}$ before $s_r^{'}$; the higher the value, the more we prefer word $s_l^{'}$ occurs before $s_r^{'}$. Whether $S_l^{'}$ should occur before $S_r^{'}$ depends on how often this reordering occurs when we reorder the source to target sentence order.

To train B, we used binary feature functions ϕ as used in (Tromble and Eisner, 2009), which were introduced for dependency parsing by McDonald et al. (2005). Table 2 shows the kind of features we used in our experiments. We did not use context features like surrounding word pos features in Table 3 because they were not useful in our preliminary experiments and propose an efficient implementation described in the next section in order to calculate this reordering model when decoding. To train the parameter θ, we used the perceptron algorithm following Tromble and Eisner (2009).

3.4 Integration to Cube Pruning

CKY parsing and cube-pruning are used for decoding of hierarchical phrase-based model (Chiang, 2007). Figure 3 displays that hierarchical phrase-based decoder seeks new span [1,7] items

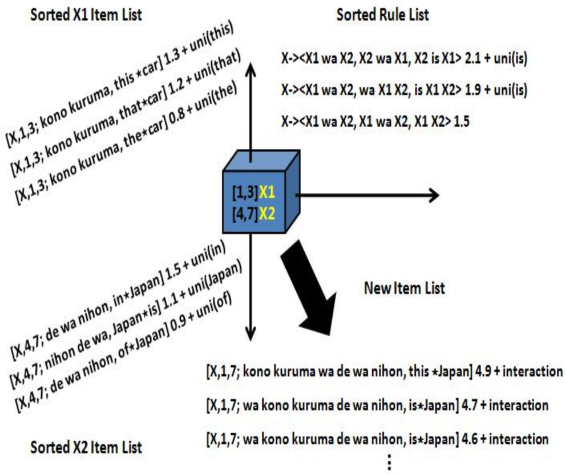


Figure 3: Creating new items from subitems and rules, that have a span [1,7] in source sentence.

with rules, utilizing subspan [1,3] items and subspan [4,7] items. In this example, we use 2-gram language model and +LM decoding. uni(・) means 1-gram language model cost for heuristics and interaction usually means language model cost that cannot be calculated offline. Here, we introduce our two implementations to calculate word-based reordering model scores in this decoding algorithm.

First, we explain a naive implementation shown in the left side of Figure 4. This algorithm performs the same calculation of reordering model as that of language model. Each item keeps a part of reordered source sentence. The reordering score of new item can be calculated as interaction cost when combining subitems with the rule.

The right side of Figure 4 shows our proposed implementation. This implementation can be adopted to decoding only when we do not use context features like surrounding word pos features in Table 3 (and consider a distance between words in features). If a span is given, the reordering scores of new item can be calculated for each rule, being independent from the word order of reordered source segment of a subitem. So, the reordering model scores can be calculated for all rules with spans by using a part of the input source sentence before sorting them for cube pruning. We expect this sorting of rules with reordering

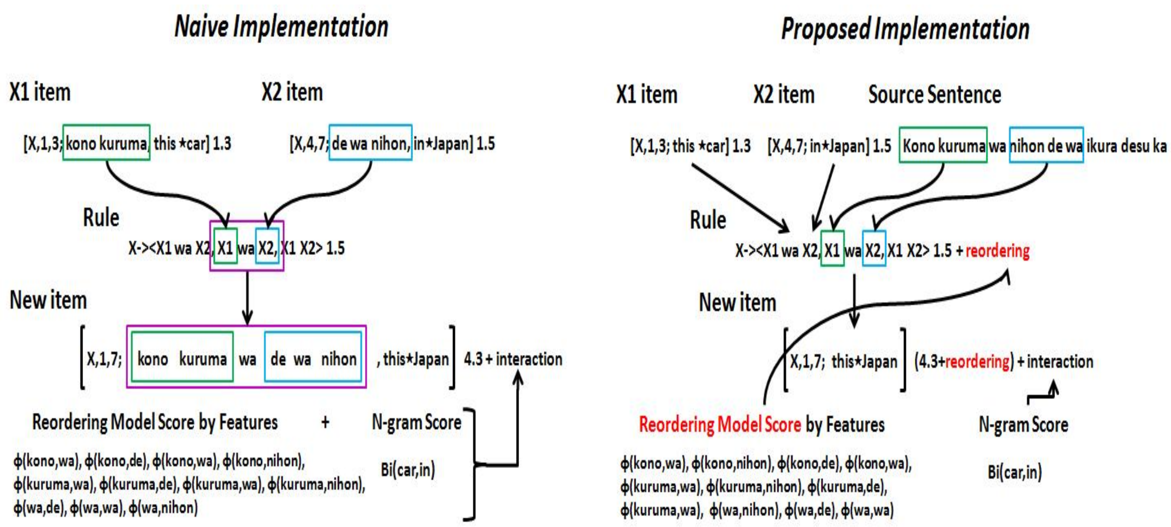


Figure 4: The "naive" and "proposed" implementation to calculate the reordering cost of new items.

model scores will have good influence on cube pruning. The right hand side of Figure 4 shows the diffrence between naive and proposed implementation (S' is not shown to allow for a clear presentation). Note the difference is in where/when the reordering scores are inserted: together with the N-gram scores in the case of naive implementation; incorpolated into sorted rules for the proposed implementation.

4 Experiment

4.1 Purpose

To reveal the effectiveness of integrating the reordering model into decoder, we compared the following setups:

- baseline: a standard hierarchical phrase-based machine translation (Hiero) system.
- preprocessing: applied Tromble and Eisner's approach, then translate by Hiero system.
- Hiero system + reordering model: integrated reordering model into Hiero system.

We used the Joshua Decoder (Li and Khudanpur, 2008) as the baseline Hiero system. This decoder uses a log-linear model with seven features, which consist of N-gram language model $P_{LM}(T)$, lexical translation model $P_w(\gamma|\alpha)$, $P_w(\alpha|\gamma)$, rule translation model $P(\gamma|\alpha)$, $P(\alpha|\gamma)$, word penalty and arity penalty.

The "Hiero + Reordering model" system has word-based reordering model as an additional feature to baseline features. For this approach, we use two systems. One has "move-to-front" system and the other is "attach" system explained in Section 3.2. We implemented our proposed algorithm in Section 3.4 to both "Hiero + Reordering model" systems. As for beam width, we use the same setups for each system.

4.2 Data Set

Data		Sent.	Word.	Avg. leng
Training	ja	200.8K	2.4M	12.0
	en	200.8K	2.3M	11.5
Development	ja	1.0K	10.3K	10.3
	en	1.0K	9.8K	9.8
Test	ja	1.0K	14.2K	14.2
	en	1.0K	13.5K	13.5

Table 4: The Data statistics

For experiments we used a Japanese-English basic travel expression corpus (BTEC). Japanese word order is linguistically very different from English and we think Japanese-English pair is a very good test bed for evaluating reordering model.

Metrics / System	BLEU	PER
Baseline (Hiero)	28.09	39.68
Preprocessing	17.32	45.27
Hiero + move-to-front	**28.85**	39.89
Hiero + attach	**29.25**	**39.43**

Table 5: BLEU and PER scores on the test set.

Our training corpus contains about 200.8k sentences. Using the training corpus, we extracted hierarchical phrase rules and trained 4-gram language model and word-based reordering model. Parameters were tuned over 1.0k sentences (development data) with single reference by minimum error rate training (MERT) (Och, 2003). Test data consisted of 1.0k sentences with single reference. Table 4 shows the condition of corpus in detail.

4.3 Results

Table 5 shows the BLEU (Papineni et al., 2001) and PER (Niesen et al., 2000) scores obtained by each system. The results clearly indicated that our proposed system with word-based reordering model (move-to-front or attach) outperformed baseline system on BLEU scores. In contrast, there is no significant improvement from baseline on PER. This suggests that the improvement of BLEU mainly comes from reordering. In our experiment, preprocessing approach resulted in very poor scores.

4.4 Discussion

Table 6 displays examples showing the cause of the improvements of our system with reordering model (attach) comparing to baseline system. We can see that the outputs of our system are more fluent than those of baseline system because of reordering model.

As a further analysis, we calculated the BLEU scores of Japanese $S^{'}$ predicted from reordering model against true Japanese $S^{'}$ made from GIZA++ alignments, were only 26.2 points on development data. We think the poorness mainly comes from unaligned words since they are untractable for the word-based reordering model. Actually, Japanese sentences in our training data include 34.7% unaligned words. In spite of the poorness, our proposed method effectively utilize this reordering model in contrast to preprocessing approach.

5 Related Work

Our approach is similar to preprocessing approach (Xia and McCord, 2004; Collins et al., 2005; Li et al., 2007; Tromble and Eisner, 2009) in that it reorders source sentence in target order. The difference is this sentence reordering is done in decoding rather than in preprocessing.

A lot of studies on lexicalized reordering (Tillman, 2004; Koehn et al., 2005; Nagata et al., 2006) focus on the phrase-based model. These works cannnot be directly applied to hierarchical phrase-based model because of the difference between normal phrases and hierarchical phrases that includes nonterminal symbols.

Shen et al. (2008,2009) proposed a way to integrate dependency structure into target and source side string on hierarchical phrase rules. This approach is similar to our approach in extending the formalism of rules on hierarchical phrase-based model in order to consider the constraint of word order. But, our approach differs from (Shen et al., 2008; Shen et al., 2009) in that syntax annotation is not necessary.

6 Conclusion and Future Work

We proposed a method to integrate word-based reordering model into hierarchical phrase-based machine translation system. We add $\gamma^{'}$ into the hiero rules, but this does not increase the number of rules. So, this extension itself does not affect the search space of decoding. In this paper we used Tromble and Eisner's reordering model for our method, but various reordering model can be incorporated to our method, for example $S^{'}$ N-gram language model. Our experimental results on Japanese-to-English task showed that our system outperformed baseline system and preprocessing approach.

In this paper we utilize $\gamma^{'}$ only for reordering model. However, it is possible to use $\gamma^{'}$ for other modeling, for example we can use it for rule translation probabilities $P(\gamma^{'}|\gamma)$, $P(\gamma|\gamma^{'})$ for additional feature functions. Of course, we can

S	america de seihin no hanbai wo hajimeru keikaku ga ari masu ka .	kono tegami wa koukuubin de nihon made ikura kakari masu ka .
T_B	sales of product in america are you planning to start ?	this letter by airmail to japan . how much is it ?
T_P	are you planning to start products in the u.s. ?	how much does it cost to this letter by airmail to japan ?
R	do you plan to begin selling your products in the u.s. ?	how much will it cost to send this letter by air mail to japan ?

Table 6: Examples of outputs for input sentence S from baseline system T_B and our proposed system (attach) T_P. R is a reference. The underlined portions have equivalent meanings and show the reordering differences.

also utilize reordered target sentence $T^{'}$ for various modeling as well. Addtionally we plan to use $S^{'}$ for MERT because we hypothesize the fluent $S^{'}$ leads to fluent T.

References

AI-Onaizan, Y. and K. Papineni. 2006. Distortion models for statistical machine translation. In *Proc. the 44th ACL*, pages 529–536.

Brown, P. F., S. A. D. Pietra, V. D. J. Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguitics*, 19:263–312.

Chiang, D., K. Knight, and W. Wang. 2009. 11,001 new features for statistical machine translation. In *Proc. NAACL*, pages 216–226.

Chiang, D. 2007. Hierachical phrase-based translation. *Computational Linguitics*, 33:201–228.

Collins, M., P. Koehn, and I. Kucerova. 2005. Clause restructuring for statistical machine translation. In *Proc. the 43th ACL*, pages 531–540.

Collins, M. 2002. Discriminative training methods for hidden markov models. In *Proc. of EMNLP*.

Freund, Y. and R. E. Schapire. 1996. Experiments with a new boosting algorithm. In *Proc. of the 13th ICML*, pages 148–156.

Koehn, P., A. Axelrod, A-B. Mayne, C. Callison-Burch, M. Osborne, and D. Talbot. 2005. Edinburgh system description for 2005 iwslt speech translation evaluation. In *Proc. the 2nd IWSLT*.

Li, Z. and S. Khudanpur. 2008. A scalable decoder for parsing-based machine translation with equivalent language model state maintenance. In *Proc. ACL SSST*.

Li, C-H., D. Zhang, M. Li, M. Zhou, K. Li, and Y. Guan. 2007. A probabilistic approach to syntax-based reordering for statistical machine translation. In *Proc. the 45th ACL*, pages 720–727.

McDonald, R., K. Crammer, and F. Pereira. 2005. Spanning tree methods for discriminative training of dependency parsers. In *Thechnical Report MS-CIS-05-11, UPenn CIS*.

Nagata, M., K. Saito, K. Yamamoto, and K. Ohashi. 2006. A clustered global phrase reordering model for statistical machine translation. In *COLING-ACL*, pages 713–720.

Niesen, S., F.J. Och, G. Leusch, and H. Ney. 2000. An evaluation tool for machine translation: Fast evaluation for mt research. In *Proc. the 2nd International Conference on Language Resources and Evaluation*.

Och, F. J. and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. the 40th ACL*, pages 295–302.

Och, F. and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29:19–51.

Och, F. J. 2003. Minimum error rate training in statistical machine translation. In *Proc. the 41th ACL*, pages 160–167.

Papineni, K. A., S. Roukos, T. Ward, and W-J. Zhu. 2001. Bleu: a method for automatic evaluation of machine translation. In *Proc. the 39th ACL*, pages 311–318.

Shen, L., J. Xu, and R. Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proc. ACL*, pages 577–585.

Shen, L., J. Xu, B. Zhang, S. Matsoukas, and R. Weischedel. 2009. Effective use of linguistic and contextual information for statistical machine translation. In *Proc. EMNLP*, pages 72–80.

Tillman, C. 2004. A unigram orientation model for statistical machine translation. In *Proc. HLT-NAACL*, pages 101–104.

Tromble, R. and J. Eisner. 2009. Learning linear ordering problems for better translation. In *Proc. EMNLP*, pages 1007–1016.

Watanabe, T., H. Tsukada, and H. Isozaki. 2006. Left-to-right target generation for hierarchical phrase-based translation. In *Proc. COLING-ACL*, pages 777–784.

Xia, F. and M. McCord. 2004. Improving a statistical mt system with automatically learned rewrite patterns. In *Proc. the 18th ICON*, pages 508–514.

A Novel Reordering Model Based on Multi-layer Phrase for Statistical Machine Translation

Yanqing He[1], Yu Zhou[2], Chengqing Zong[2], Huilin Wang[1]
[1]Institute of Scientific and Technical Information of China
{heyq,wanghl}@istic.ac.cn
[2]Institute of Automation, Chinese Academy of Sciences
{yzhou,cqzong}@nlpr.ia.ac.cn

Abstract

Phrase reordering is of great importance for statistical machine translation. According to the movement of phrase translation, the pattern of phrase reordering can be divided into three classes: monotone, BTG (Bracket Transduction Grammar) and hierarchy. It is a good way to use different styles of reordering models to reorder different phrases according to the characteristics of both the reordering models and phrases itself. In this paper a novel reordering model based on multi-layer phrase (PRML) is proposed, where the source sentence is segmented into different layers of phrases on which different reordering models are applied to get the final translation. This model has some advantages: different styles of phrase reordering models are easily incorporated together; when a complicated reordering model is employed, it can be limited in a smaller scope and replaced with an easier reordering model in larger scope. So this model better trade-offs the translation speed and performance simultaneously.

1 Introduction

In statistical machine translation (SMT), phrase reordering is a complicated problem. According to the type of phrases, the existing phrase reordering models are divided into two categories: contiguous phrase-based reordering models and non-contiguous phrase-based reordering models.

Contiguous phrase-based reordering models are designed to reorder contiguous phrases. In such type of reordering models, a contiguous phrase is reordered as a unit and the movements of phrase don't involve insertions inside the other phrases. Some of these models are content-independent, such as distortion models (Och and Ney, 2004; Koehn et al., 2003) which penalize translation according to jump distance of phrases, and flat reordering model (Wu, 1995; Zens et al., 2004)which assigns constant probabilities for monotone order and non-monotone order. These reordering models are simple and the contents of phrases have not been considered. So it's hard to obtain a satisfactory translation performance. Some lexicalized reordering models (Och et al., 2004; Tillmann 2004, Kumar and Byrne, 2005, Koehn et al., 2005) learn local orientations (monotone or non-monotone) with probabilities for each bilingual phrase from training data. These models are phrase-dependent, so improvements over content-independent reordering models are obtained. However, many parameters need to be estimated.

Non-contiguous phrase-based reordering models are proposed to process non-contiguous phrases and the movements of phrase involve insertion operations. This type of reordering models mainly includes all kinds of syntax-based models where more structural information is employed to obtain a more flexible phrase movement. Linguistically syntactic approaches (Yamada and Knight, 2001; Galley et al., 2004, 2006; Marcu et al., 2006; Liu et al., 2006; Shieber et al., 1990; Eisner, 2003; Quirk et al., 2005; Ding and Palmer, 2005) employ linguistically syntactic information to enhance their reordering capability and use non-contiguous phrases to

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 447–455,
Beijing, August 2010

obtain some generalization. The formally syntax-based models use synchronous context-free grammar (SCFG) but induce a grammar from a parallel text without relying on any linguistic annotations or assumptions (Chiang, 2005; Xiong et al., 2006). A hierarchical phrase-based translation model (HPTM) reorganizes phrases into hierarchical ones by reducing sub-phrases to variables (Chiang 2005). Xiong et al. (2006) is an enhanced bracket transduction grammar with a maximum entropy-based reordering model (MEBTG). Compared with contiguous phrase-based reordering model, Syntax-based models need to shoulder a great deal of rules and have high computational cost of time and space. The type of reordering models has a weaker ability of processing long sentences and large-scale data, which heavily restrict their application.

The above methods have provided various phrases reordering strategies. According to the movement of phrase translation, the pattern of phrase reordering can be divided into three classes: monotone, BTG (Bracket Transduction Grammar) (Wu, 1995) and hierarchy. In fact for most sentences, there may be some phrases which have simple reordering patterns, such as monotone or BTG style. It is not necessary to reorder them with a complicated mechanism, e.g. hierarchy. It is a good idea that different reordering models are employed to reorder different phrases according to the characteristics of both the reordering models and the phrases itself. This paper thus gives a novel reordering model based on multi-layer phrase (PRML), where the source sentence is segmented into different layers of phrases on which different reordering models are applied to get the final translation. Our model has the advantages as follow: (1) PRML segments source sentence into multiple-layer phrases by using punctuation and syntactic information and the design of segmentation algorithm corresponds to each reordering model. Different reordering models are chosen for each layer of phrases. (2) In our model different reordering models can be easily integrated together to obtain a combination of multiple phrase reordering models. (3) Our model can incorporate some complicated reordering models. We limit them in relatively smaller scopes and replace them with easier reordering models in larger scopes. In such way our model better trade-offs the translation speed and performance simultaneously. (4) Our segmentation strategy doesn't impair translation quality by controlling phrase translation tables to determine the scope of each reordering model in each source sentence. The poor phrase translations generated by the former reordering model, still have chances of being revised by the latter reordering model.

Our work is similar to the phrase-level system combination (Mellebeek et al., 2006). We share one important characteristic: we decompose input sentence into chunks and recompose the translated chunks in output. The differences are that, we segment the input sentence into multi-layer phrases and we reorder their translations with a multi-layer decoder.

The remainder of the paper is organized as follows: Section 2 gives our reordering model PRML. Section 3 presents the details of the sentence segmentation algorithm and the decoding algorithm. Section 4 shows the experimental results. Finally, the concluding remarks are given in Section 5.

2 The Model

We use an example to demonstrate our motivation. Figure 1 shows a Chinese and English sentence pair with word alignment. Each solid line denotes the corresponding relation between a Chinese word and an English word. Figure 2 shows our reordering mechanism. For the source sentence, the phrases in rectangle with round corner in row 2 obviously have a monotone translation order. For such kinds of phrase a monotone reordering model is enough to arrange their translations. Any two neighbor consecutive phrases in the ellipses in row 3 have a straight orders or inverted order. So BTG reordering model is appropriate to predict the order of this type of phrases. Inside the phrases in the ellipses in row 3 there are possibly more complicated hierarchical structures. For the phrase "*通往 和平 之 路*", a rule "$X \rightarrow \langle$通往 X_1 之路, *towards the road to* $X_1 \rangle$" has the ascendancy over the monotone and BTG style of reordering model. Hierarchy style of reordering models, such as HPTM reordering model, can translate non-contiguous phrases and has the advantage of capturing the translation of such kind of phrases.

The whole frame of our model PRML is shown in Figure 3. PRML is composed of a segmentation sentence module and a decoder which consists of three different styles of phrase reordering models. The source sentence is segmented into 3 layers of phrases: the original whole sentence, sub-sentences and chunks. The original whole sentence is considered as the first-layer phrase and is segmented into sub-sentences to get the second-layer phrase. By further segmenting these sub-sentences, the chunks are obtained as the third-layer phrase. The whole translation process includes three steps: **1)** In order to capture the most complicated structure of phrases inside chunks, HPTM reordering model are chosen to translate the chunks. So the translations of chunks are obtained. 2) Combine the bilingual chunks generated by step 1 with those bilingual phases generated by the MEBTG training model as the final phrase table and translate the sub-sentences with MEBTG reordering model, the translations of sub-sentences are obtained. 3) Combine the bilingual sub-sentences generated by step 2 with those bilingual phases generated by the Monotone training model as the final phrase table and translate the original whole sentences with monotone reorder-

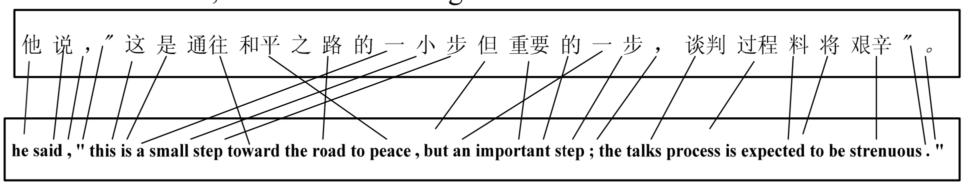


Figure 1. An example of Chinese-English sentence pair with their word alignment

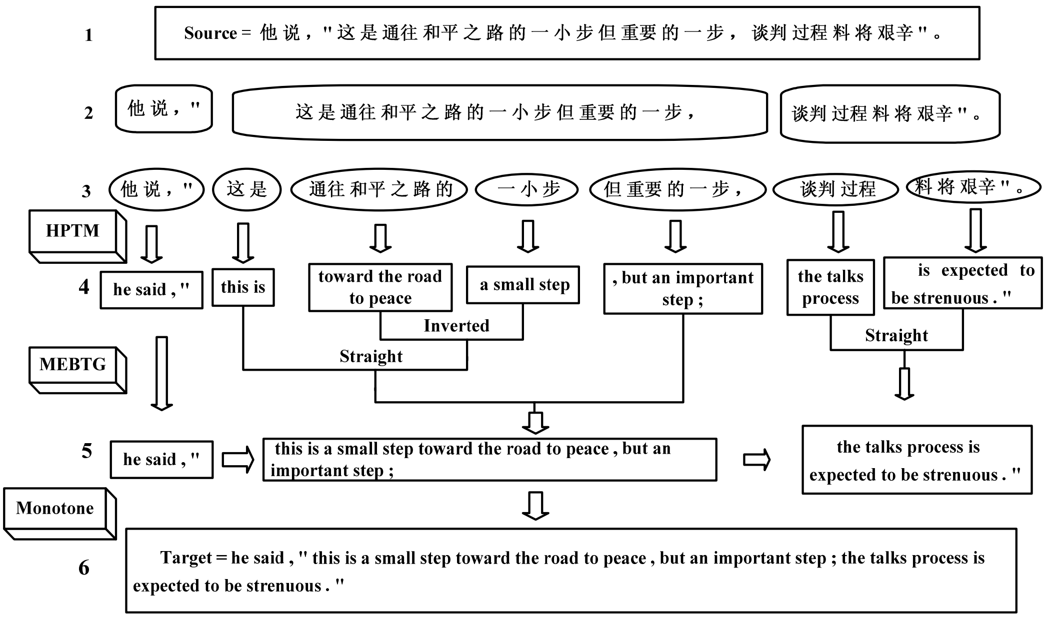


Figure 2. Diagram of Translation Using PRML.

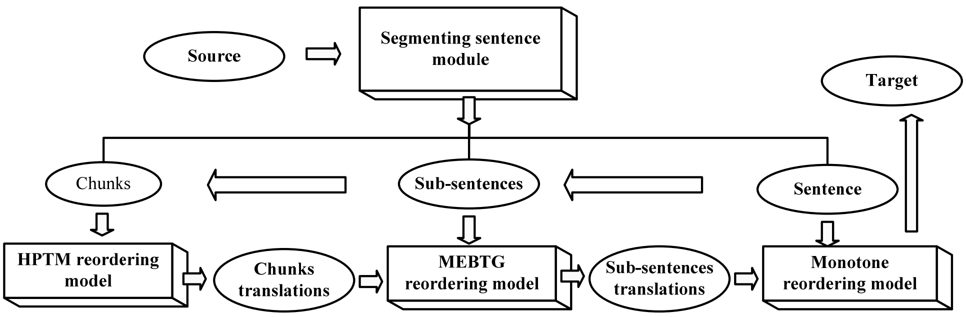


Figure 3. Frame of PRML

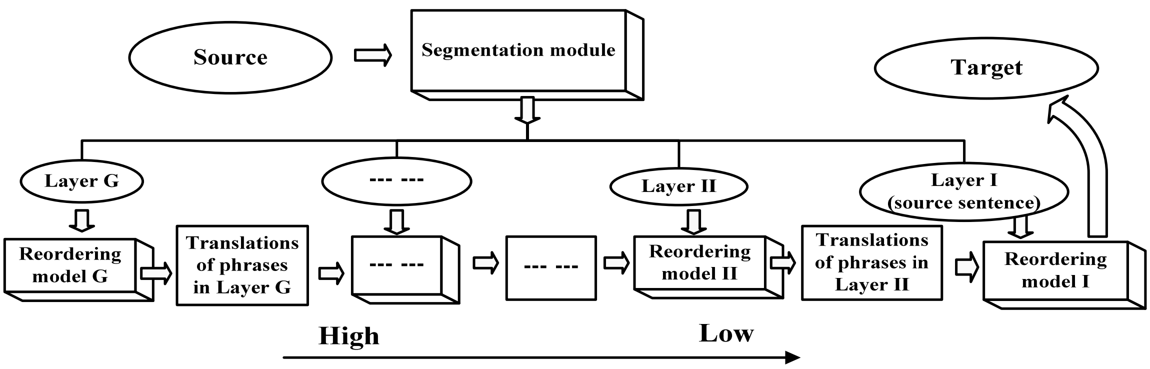


Figure 4. General frame of our model

ing model, the translations of the original whole sentences are obtained.

We also give a general frame of our model in Figure 4. In the segmentation module, an input source sentence is segmented into *G* layers of contiguous source strings, Layer 1, Layer 2, …, Layer *G*. The phrases of lower-order layer are re-segmented into the phrases of higher-order layer. The phrases of the same layer can be combined into the whole source sentence. The decoding process starts from the phrases of the highest-order layer. For each layer of phrases a reordering model is chosen to generate the translations of phrases according to their characteristics. The generated translations of phrases in the higher-order layer are fed as a new added translation source into the next lower-order reordering model. After the translations of the phrase in Layer 2 are obtained, they are fed into the Reordering model 1 as well as the source sentence (the phrase in Layer 1) to get the target translation.

Due to the complexity of the language, there may be some sentences whose structures don't conform to the pattern of the reordering models we choose. So in our segmentation module, if the sentence doesn't satisfy the segmentation conditions of current layer, it will be fed into the segmentation algorithm of the next layer. Even in the worst condition when the sentence isn't segmented into any phrase by segmentation module, it will be translated as the whole sentence to get the final translation by the highest-order reordering model.

Our model tries to grasp firstly the simple reordering modes in source sentence by the lower layer of phrase segmentations and controls more complicated reordering modes inside the higher layers of phrases. Then we choose some complicated reordering models to translate those phrases. Thus search space and computational complexity are both reduced. After obtaining the translation of higher layer's phrases, it is enough for simple reordering models to reorder them. Due to phrase segmentation some phrases may be translated poorly by the higher layer of reordering models, but they still have chances of being revised by the lower layer of reordering model because in lower layer of reordering model the input phrases have not these hard segmentation boundary and our model uses phrase translation tables to determine the scope of each reordering model.

There are two key issues in our model. The first one is how to segment the source sentence into different layers of phrases. The second one is how to choose a reordering model for different layer of phrases. In any case the design of segmenting sentence module should consider the characteristic of the reordering model of phrases.

3 Implementation

The segmentation module consists of the sub-sentence segmentation and chunk segmentation. The decoder combines three reordering models, HPTM, MEBTG, and a monotone reordering model.

3.1 Segmentation module

We define the sub-sentence as the word sequence which can be translated in monotone order. The following six punctuations: 。 ！ ？ ， ： ； in Chinese, and . ! ? , : ; in English are chosen as the segmentation anchor candidates. Except Chinese comma, all the other five punctuations can express one semantic end and another semantic beginning. In most of the time, it has high error risk to segment the source sentence by commas. So we get help from syntactic information of Chinese dependency tree to guarantee the monotone order of Chinese sub-sentences.

The whole process of sub-sentence segmentation includes training and segmenting.

Training: 1) The word alignment of training parallel corpus is obtained; **2)** The parallel sentence pairs in training corpus are segmented into sub-sentences candidates. For a Chinese-English sentence pair with their word alignment in training data, all bilingual punctuations are found firstly, six punctuations respectively "？！。，：；" in Chinese and "? ! . , : ;" in English. The punctuation identification number (id) sets in Chinese and English are respectively extracted. For a correct punctuation id pair (*id_c*, *id_e*), the phrase before *id_e* in English sentence should be the translation of the phrase before *id_c* in Chinese sentence, namely the number of the links[1] between the two phrases should be equal. In order to guarantees the property we calculate a bilingual alignment ratio for each Chinese-English punctuation id pair according to the following equation. For the punctuation id pair (*id_c*, *id_e*), bilingual alignment ratio consists of two value, Chinese-English alignment ratio (*CER*) and English-Chinese alignment ratio (*ECR*).

$$CER = \frac{\sum_{\substack{1\le i\le id\_c \\ 1\le j\le J}} \delta(A_{ij})}{\sum_{\substack{1\le j\le id\_e \\ 1\le i\le I}} \delta(A_{ij})} \qquad ECR = \frac{\sum_{\substack{1\le j\le id\_e \\ 1\le i\le I}} \delta(A_{ij})}{\sum_{\substack{1\le i\le id\_c \\ 1\le j\le J}} \delta(A_{ij})}$$

where $\delta(A_{ij})$ is an indicator function whose value is 1 when the word id pair (i, j) is in the word alignment and is 0 otherwise. I and J are the length of the Chinese English sentence pair. *CER* of a correct punctuation id pair will be equal to 1.0. So does *ECR*. In view of the error rate of word alignment, the punctuation id pairs will be looked as the segmentation anchor if both *CER* and *ECR* are falling into the threshold range (*minvalue*, *maxvalue*). Then all the punctuation id pairs are judged according to the same method and those punctuation id pairs satisfying the requirement segment the sentence pair into sub-sentence pairs. **3)** The first word of Chinese sub-sentence in each bilingual sub-sentence pair is collected. We filter these words whose frequency is larger than predefined threshold to get segmentation anchor word set (*SAWS*).

Segmenting: 1) The test sentence in Chinese is segmented into segments by the six Chinese punctuation "。！？，：；" in the sentence. **2)** If the first word of a segment is in *SAWS* the punctuation at the end of the segment is chosen as the segmentation punctuation. **3)** If a segment satisfies the property of "dependency integrity" the punctuation at the end of the segment is also chosen as the segmentation punctuation. Here "dependency integrity" is defined in a dependency tree. Figure 5 gives the part output

ID	word	POS	head id	dependency type
1	美国	NR	3	NMOD
2	国会	NN	3	NMOD
3	议员	NN	4	SUB
4	表示	VV	0	ROOT
5	，	PU	4	P
6	人民币	NN	7	VMOD
7	低估	VV	9	VMOD
8	，	PU	9	P
……	……	……	……	……

Figure 5. The part dependency parser output of a Chinese sentence.

of "lexical dependency parser"[2] for a Chinese sentence. There are five columns of data for each word which are respectively the word id, the word itself, its speech of part, the id of its head word and their dependency type. In the sentence the Chinese word sequence "*美国 国会 议员 表示* (US congressional representatives say that)" has such a property: Each word in the sequence has a dependency relation with the word which is still in the sequence except one word which has a dependency relation with the root, e.g. id 4. We define the property as "dependency integrity". Our reason is: a sub-sentence with the property of "dependency integrity" has relatively independent semantic meaning and a large possibility of monotone translation order. **4)** The union of the segmentation punctuations in step 2) and 3) are the final sub-sentence segmentation tags.

[1] Here a link between a Chinese word and an English word means the word alignment between them.

[2] http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html

After sub-sentence segmentation, chunks segmentation is carried out in each sub-sentence. We define the chunks as the word sequence which can be translated in monotone order or inverted order. Here the knowledge of the "phrase structure parser"[3] and the "lexicalized dependency parser" are integrated to segment the sub-sentence into chunks. In a Chinese phrase structure parser tree the nouns phrase (NP) and preposition phrase (PP) are relatively independent in semantic expressing and relatively flexible in translation. So in the chunk segmentation, only the NP structure and PP structure in the Chinese structure parsing tree are found as phrase structure chunk. The process of chunk segmentation is described as follows: **1)** the test sub-sentence is parsed to get the phrase structure tree and dependency parsing tree; **2)** We traverse the phrase structure tree to extract sub-tree of "NP" and "PP" to obtain the phrase structure chunks. **3)** We mark off the word sequences with "dependency integrity" in the dependency tree. **4)** Both the two kinds of chunks are recombined to obtain the final result of chunk segmentation.

3.2 Decoding

Our decoder is composed of three styles of reordering models: HPTM, MEBTG and a monotone reordering model.

According to Chiang (2005), given the chunk c_{chunk}, a CKY parser finds $\widehat{e}_{chunk}$, the English yield of the best derivation $\widehat{D}_{hptm}$ that has Chinese yield c_{chunk}:

$$\begin{aligned}\widehat{e}_{chunk} &= e_{chunk}(\widehat{D}_{hptm}) \\ &= e_{chunk}(\operatorname*{arg\,max}_{C(D_{hptm})=C_{chunk}} \Pr(D_{hptm}))\end{aligned}$$

Here the chunks not the whole source sentence are fed into HPTM decoder to get the L-best translations and feature scores of the chunks. We combine all the chunks, their L-best translations and the feature scores into a phrase table, namely chunk phrase table. We only choose 4 translation scores (two translation probability based on frequency and two lexical weights based on word alignment) because the language model score, phrase penalty score and word penalty score will be re-calculated in the lower layer of reordering model and need not be kept here. Meantime we change the log values of the scores into probability value. In the chunk phrase table each phrase pair has a Chinese phrase, an English phrase and four translations feature scores. In each phrase pair the Chinese phrase is one of our chunks, the English phrase is one translation of L-best of the chunk.

In MEBTG (Xiong et al., 2006), three rules are used to derive the translation of each sub-sentence: lexical rule, straight rule and inverted rule. Given a source sub-sentence $C_{sub-sent}$, it finds the final sub-sentence translation $\widehat{E}_{sub-sent}$ from the best derivation $\widehat{D}_{mebtg}$:

$$\begin{aligned}\widehat{E}_{sub-sent} &= E_{sub-sent}(\widehat{D}_{mebtg}) \\ &= E(\operatorname*{arg\,max}_{C(D_{mebtg})=C_{sub-sent}} \Pr(D_{mebtg}))\end{aligned}$$

Generally chunk segmentation will make some HPTM rules useless and reduce the translation performance. So in MEBTG we also use base phrase pair table which contains the contiguous phrase translation pairs consistent with word alignment. We merge the chunk phrase table and base phrase table together and feed them into MEBTG to translate each sub-sentence. Thus the K-Best translation and feature scores of each sub-sentence are obtained and then are recombined into a new phrase table, namely sub-sentence phrase table, by using the same method with chunk phrase table.

Having obtained the translation of each sub-sentence we generate the final translation of the whole source sentence by a monotone reordering model. Our monotone reordering model employs a log-linear direct translation model. Three phrase tables: chunk phrase table, sub-sentence phrase table and base phrase table are merged together and fed into the monotone decoder. Thus the decoder will automatically choose those phrases it need. In each phrase table each source phrase only has four translation probabilities for its candidate translation. So it's easy to merge them together. In such way all kinds of phrase pairs will automatically compete according to their translation probabilities. So our PRML model can automatically decide which reordering model is employed in each phrase scope of the whole source sentence. It's worth noting that the inputs of the three reordering

[3] http://nlp.stanford.edu/software/lex-parser.shtml

model have no segmentation tag. Because any segmentation for the input before decoding will influence the use of some rules or phrase pairs and may cause some rules or phrase pairs losses. It would be better to employ different phrase table to limit reordering models and let each decoder automatically decide reordering model for each segments of the input. Thus by controlling the phrase tables we apply different reordering models on different phrases. For each reordering model we perform the maximum BLEU training (Venugopal et al. 2005) on a development set. For HPTM the training is same as Chiang 2007. For MEBTG we use chunk phrase table and base table to obtain translation parameters. For monotone reordering model all the three phrase tables are merged to get translation weights.

4 Experiments

This section gives the experiments with Chinese-to-English translation task in news domain. Our evaluation metric is case-insensitive BLEU-4 (Papineni et al. 2002). We use NIST MT 2005, NIST MT 2006 and NIST MT 2008 as our test data. Our training data is filtered from the LDC corpus[4]. Table 1 gives the statistics of our data.

Set	Language	Sentence	Vocabulary	A. S. L
Train data	Chinese	297,069	6,263	11.9
	English	297,069	8,069	13.6
NIST 05	Chinese	1,082	5669	28.2
	English	4,328	7575	32.7
NIST 06	Chinese	1,664	6686	23.5
	English	6,656	9388	28.9
NIST 08	Chinese	1,357	6,628	24.5
	English	5,428	9,594	30.8

Table 1. The statistics of training data and test data, A. S. L is average sentence length.

4.1 Evaluating translation Performance

We compare our PRML against two baselines: MEBTG system developed in house according to Xiong (2006, 2008) and HPTM system[5] in PYTHON based on HPTM reordering model (Chiang 2007). In MEBTG phrases of up to 10 words in length on the Chinese side are extracted and reordering examples are obtained without limiting the length of each example. Only the last word of each reordering example is used as lexical feature in training the reordering model by the maximum entropy based classifier[6]. We also set a swapping window size as 8 and the beam threshold as 10. It is worth noting that our MEBTG system uses cube-pruning algorithm (Chiang 2005) from bottom to up to generate the *N*-best list not the lazy algorithm of (Huang and Chiang, 2005). We also limit the length of the HPTM initial rules no more than 10 words and the number of non-terminals within two. In the decoding for the rules the beam pruning parameter is 30 and threshold pruning parameter is 1.0. For hypotheses the two pruning parameters are respectively 30 and 10. In our PRML *minvalue*=0.8, *maxvalue*=1.25, which are obtained by minimum error rate training on the development set. The predefined value for filtering *SAWS* is set as 100.

The translation performance of the three reordering model is shown in Table 2. We can find that PRML has a better performance than MEBTG with a relatively 2.09% BLEU score in NIST05, 5.60% BLEU score in NIST06 and 5.0% BLEU score in NIST08. This indicates that the chunk phrase table increases the reordering ability of MEBTG. Compared with HPTM, PRML has a comparable translation performance in NIST08. In NIST05 and NIST06 our model has a slightly better performance than HPTM. Because PRML limit hierarchical structure reordering model in chunks while HPTM use them in the whole sentence scope (or in a length scope), HPTM has a more complicated reordering mechanism than PRML. The experiment result shows even though we use easier reordering moels in larger scope, e.g. MEBTG and monoto-

Model	Nist05	Nist06	Nist08
HPTM	0.3183	0.1956	0.1525
MEBTG	0.3049	0.1890	0.1419
PRML	0.3205	0.1996	0.1495

Table 2. The translation performance

[4] LDC corpus lists: LDC2000T46, LDC2000T50, LDC2002E18, LDC2002E27, LDC2002L27, LDC2002T01, LDC2003E07, LDC2003E14, LDC2003T17, LDC2004E12, LDC2004T07, LDC2004T08, LDC2005T01, LDC2005T06, LDC2005T10, LDC2005T34, LDC2006T04, LDC2007T09

[5] We are extremely thankful to David Chiang who originally implement the PYTHON decoder and share with us.

[6] http://maxent.sourceforge.net/

ne reordering model, we have a comparatively translation performance as HPTM.

4.2 Evaluating translation speed

Table 3 shows the average decoding time on test data for the three phrase reordering models on a double processor of a dual 2.0 Xeon machine. Time denotes mean time of per-sentence, in seconds. It is seen that PRML is the slower than MEBTG but reduce decoding time with a relatively 54.85% seconds in NIST05, 75.67% seconds in NIST06 and 65.28% seconds in NIST08. For PRML, 93.65% average decoding time in NIST05 is spent in HPTM, 4.89% time in MEBTG and 1.46% time in monotone reordering decoder.

Model	Nist05	Nist06	Nist08
HPTM	932.96	1235.21	675
MEBTG	43.46	27.16	10.24
PRML	421.20	300.52	234.33

Table 3. The average decoding time

4.3 Evaluating the performance of each layer of phrase table

In order to evaluate the performance of each reordering model, we run the monotone decoder with different phrase table in NIST05. Table 4 list the size of each phrase table. From the results in Table 5 it is seen that the performance of using three phrase tables is the best. Compared with the base phrase table, the translation performances are improved with relatively 10.86% BLEU score by adding chunk phrase table and 11% BLEU score by adding sub-sentence table. The result of row 4 has a comparable to the one in row 5. It indicates the sub-sentence phrase table has contained the information of HPTM reordering model. The case of row 4 to row 2 is the same.

Phrase table	Phrase pair
Base	732732
Chunk	86401
Sub-sentence	24710

Table 4. The size of each phrase table.

Phrase table	Reordering model	BLEU
Base	Monotone	0.2871
Base +chunk	monotone+HPTM	0.3180
Base +sub-sentence table	monotone+HPTM +MEBTG	0.3187
Base +chunk +subsentence	monotone+HPTM +MEBTG	0.3205

Table 5. The performance of phrase table

5 Conclusions

In this paper, we propose a novel reordering model based on multi-layer phrases (PRML), where the source sentence is segmented into different layers of phrases and different reordering models are applied to get the final translation. Our model easily incorporates different styles of phrase reordering models together, including monotone, BTG, and hierarchy or other more complicated reordering models. When a complicated reordering model is used, our model can limit it in a smaller scope and replace it with an easier reordering model in larger scope. In such way our model better trade-offs the translation speed and performance simultaneously.

In the next step, we will use more features to segment the sentences such as syntactical features or adding a dictionary to supervise the segmentation. And also we will try to incorporate other systems into our model to improve the translation performance.

6 Acknowledgements

The research work has been partially funded by the Natural Science Foundation of China under Grant No. 6097 5053, and 60736014, the National Key Technology R&D Program under Grant No. 2006BAH03B02, the Hi-Tech Research and Development Program ("863" Program) of China under Grant No. 2006AA010108-4, and also supported by the China-Singapore Institute of Digital Media (CSIDM) project under grant No. CSIDM-200804, and Research Project "Language and Knowledge Technology" of Institute of Scientific and Technical Information of China (2009DP01-6).

References

David Chiang. 2005. *A hierarchical phrase-based model for statistical machine translation*. In Proceedings of ACL 2005, pages 263–270.

David Chiang, 2007. *Hierarchical Phrase-based Translation*. Computational Linguistics,33(2):201-228.

Yuan Ding and Martha Palmer. 2005. *Machine translation using probabilistic synchronous dependency insertion grammars*. In proceeding of 43th Meeting of the Association for Computational Linguistics, 541-548

Jason Eisner. 2003. *Learning non-isomorphic tree mappings for machine translation*. In proceedings of the 41th Meeting of the Association for Computational Linguistics (companion volume).

Michel Galley, Mark Hopkins, Kevin Knight and Daniel Marcu. 2004. *What's in a translation rule?* In proceedings of HLTNAACL- 2004.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, Ignacio Thayer. 2006. *Scalable Inference and Training of Context-Rich Syntactic Translation Models.* In Proceedings of the joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics. Sydney, Australia.

Liang Huang and David Chiang. 2005. *Better k-best parsing*. In Proceedings of the Ninth International Workshop on Parsing Technology, Vancouver, October, pages 53–64.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu, 2002. *BLEU: a method for automatic evaluation of machine translation*. In Proceedings of the 40th Annual Meeting of the ACL. page 311-318, Philadelphia, PA.

Philipp Koehn, Franz J. Och and Daniel Marcu. 2003. *Statistical phrase-based translation*. In proceedings of HLT-NAACL-03, 127-133

Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne and David Talbot. 2005. *Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation*. In International Workshop on Spoken Language Translation.

Shankar Kumar and William Byrne. 2005. *Local phrase reordering models for statistical machine translation*. In Proceedings of HLT-EMNLP.

Yang Liu, Qun Liu and Shouxun Lin. 2006. *Tree-to-String Alignment Template for Statistical Machine Translation.* In proceedings of ACL-06, 609-616.

Daniel Marcu and William Wong. 2002. *A phrase-based, joint probability model for statistical machine translation.* In proceedings of EMNLP-02, 133-139.

Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. *SPMT: Statistical Machine Translation with Syntactified Target Language Phrases*. In Proceedings of EMNLP-2006, 44-52, Sydney, Australia

Bart Mellebeek, Karolina Owczarzak, Josef Van Genabith, Andy Way. 2006. *Multi-Engine Machine Translation By Recursive Sentence Decomposition*. In Proceedings of AMTA 2006

Franz J. Och and Hermann Ney. 2004. *The alignment template approach to statistical machine translation.* Computational Linguistics, 30(4):417-449

Franz Josef Och, Ignacio Thayer, Daniel Marcu, Kevin Knight, Dragos Stefan Munteanu, Quamrul Tipu, Michel Galley, andMark Hopkins. 2004. *Arabic and Chinese MT at USC/ISI*. Presentation given at NIST Machine Translation Evaluation Workshop.

Chris Quirk, Arul Menezes and Colin Cherry. 2005. *Dependency treelet translation: Syntactically informed phrasal SMT*. In proceedings of the 43th Meeting of the Association for Computational Linguistics, 271-279

S. Shieber and Y. Schabes. 1990. *Synchronous tree adjoining grammars*. In proceedings of COLING-90.

Christoph Tillmann. 2004. *A block orientation model for statistical machine translation*. In HLT-NAACL, Boston, MA, USA.

Ashish Venugopal, Stephan Vogel and Alex Waibel. 2003. *Effective Phrase Translation Extraction from Alignment Models*, in Proceedings of the 41st ACL, 319-326.

Dekai Wu. 1995. *Stochastic inversion transduction grammars, with application to segmentation, bracketing, and alignment of parallel corpora.* In proceeding of IJCAL 1995, 1328-1334,Montreal, August.

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. *Maximum Entropy Based phrase reordering model for statistical machine translation*. In proceedings of COLING-ACL, Sydney, Australia.

Deyi Xiong, Min Zhang, Ai Ti Aw, Haitao Mi, Qun Liu and Shouxun Lin. *Refinements in BTG-based Statistical Machine Translation.* In Proceedings of IJCNLP 2008.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In proceedings of the 39th Meeting of the ACL, 523-530.

R. Zens, H. Ney, T. Watanabe, and E. Sumita. 2004. *Reordering Constraints for Phrase-Based Statistical MachineTranslation.* In Proceedings of CoLing 2004, Geneva, Switzerland, pp. 205-211.

Standardizing Wordnets in the ISO Standard LMF: Wordnet-LMF for GermaNet

Verena Henrich
Universtiy of Tübingen
Department of Linguistics
verena.henrich@uni-tuebingen.de

Erhard Hinrichs
Universtiy of Tübingen
Department of Linguistics
erhard.hinrichs@uni-tuebingen.de

Abstract

It has been recognized for quite some time that sustainable data formats play an important role in the development and curation of linguistic resources. The purpose of this paper is to show how GermaNet, the German version of the Princeton WordNet, can be converted to the Lexical Markup Framework (LMF), a published ISO standard (ISO-24613) for encoding lexical resources. The conversion builds on Wordnet-LMF, which has been proposed in the context of the EU KYOTO project as an LMF format for wordnets. The present paper proposes a number of crucial modifications and a set of extensions to Wordnet-LMF that are needed for conversion of wordnets in general and for conversion of GermaNet in particular.

1 Introduction

It has been recognized for quite some time that sustainable data formats play an important role in the development and curation of linguistic resources. As witnessed by the success of the guidelines of the Text Encoding Initiative[1] (TEI) and of published standards issued by the International Standards Organization[2] (ISO), markup languages such as XML[3] (short for: Extensible Markup Language) have become lingua francas for encoding linguistic resources of different types, including phonetic transcriptions, (annotated) text corpora, and dictionaries. It is fair to say that it has become common practice among developers of new linguistic resources to consult TEI guidelines and ISO standards in order to develop standard-conformant encoding schemes that serve as an interchange format and that can be documented and validated by Document Type Definitions (DTD) and XML schemata.

However, for resources that were developed prior to or largely in parallel with the emerging acceptance of markup languages and of emerging encoding standards, the situation is far more heterogeneous. A wide variety of legacy formats exists, many of which have persisted due to existing user communities and the availability of tools that can process only such idiosyncratic formats. The development of wordnets for a large number of languages is a typical example of a type of linguistic resource, where legacy formats still persist as a de facto standard. WordNet 1.6 is encoded in the data format of lexicographer files[4] that was designed for the English Princeton WordNet (Fellbaum, 1998). It is a plain-text format for storing wordnet data and allows lexicographers to encode lexical and conceptual relations among lexical units and synsets by use of special-purpose diacritics. There exist numerous tools that can process WordNet 1.6 lexicographer files to extract relevant information or to transform the data into other special-purpose formats such as Prolog-fact databases. Even tough still widely used for the reasons just mentioned, the complexity of the format itself has a number of undesirable consequences. As Henrich and Hinrichs (2010) have pointed out,

[1] See http://www.tei-c.org
[2] See http://www.iso.org
[3] See http://www.w3.org/TR/REC-xml/

[4] See http://wordnet.princeton.edu/man/lexnames.5WN.html

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 456–464,
Beijing, August 2010

the editing of lexicographer files is highly error-prone and time-consuming in actual lexicographic development. Moreover, format validation of the data as well as development of new tools for data visualization and data extraction become increasingly difficult since they cannot be based on generic state-of-the-art tools, that are, for example, available for XML-based encodings.

For exactly these reasons, XML-based interchange formats have been proposed in recent years also for wordnets. One of the first, if not the first, example is the XML format for GermaNet[5], a wordnet for German (Lemnitzer and Kunze, 2002; Henrich and Hinrichs, 2010). An even more recent development along these lines is the specification of Wordnet-LMF (see Soria et al., 2009), an instantiation of the Lexical Markup Framework[6] (LMF, (Francopoulo et al., 2006)) customized for wordnets.

Since LMF is an ISO standard (ISO-24613), it is a particularly attractive candidate for encoding wordnets. Everything else being equal, ISO standards have a high chance of being adopted by a wide user community and of being recognized as an interchange format.[7] Such agreed-upon interchange formats are a crucial prerequisite for interoperable linguistic resources in the context of web services and of processing pipelines for linguistic resources.

The purpose of this paper is threefold:

1. To compare and contrast the GermaNet XML initially proposed by Lemnitzer and Kunze (2002) with the Wordnet-LMF. This comparison is instructive since it reveals two completely different conceptions of representing semantic knowledge at the lexical level.
2. To point out a number of open issues that need to be resolved if Wordnet-LMF is to be adopted widely among wordnets for a steadily increasing number of languages.
3. To show how these open issues can be resolved in a customized version of Wordnet-LMF suitable for GermaNet.

The remainder of this paper is structured as follows: section 2 provides a general introduction to GermaNet. Details about the adapted XML format used for GermaNet up until now are provided in section 3. Section 4 introduces the challenge of how to represent a wordnet in the Lexical Markup Framework. As one possibility, Wordnet-LMF is regarded. Issues that arise during the conversion of GermaNet into Wordnet-LMF lead to a modified version of Wordnet-LMF. Finally, section 5 concludes with a comparison of the two representation formats.

2 GermaNet

GermaNet is a lexical semantic network that is modeled after the Princeton WordNet for English. It partitions the lexical space into a set of concepts that are interlinked by semantic relations. A semantic concept is modeled by a *synset*. A synset is a set of words (called *lexical units*) where all the words are taken to have (almost) the same meaning. Thus a synset is a set-representation of the semantic relation of synonymy, which means that it consists of a list of lexical units and a paraphrase (represented as a string). The lexical units in turn have frames (which specify the syntactic valence of the lexical unit) and examples. The list of lexical units for a synset is never empty, but any of the other properties may be.

There are two types of semantic relations in GermaNet: *conceptual* and *lexical relations*. Conceptual relations hold between two semantic concepts, i.e. synsets. They include relations such as hyperonymy, part-whole relations, entailment, or causation. Lexical relations hold between two individual lexical units. Antonymy, a pair of opposites, is an example of a lexical relation.

GermaNet covers the three word categories of adjectives, nouns, and verbs, each of which is hierarchically structured in terms of the hyperonymy relation of synsets.

[5] See http://www.sfs.uni-tuebingen.de/GermaNet/
[6] See http://www.lexicalmarkupframework.org
[7] An anonymous reviewer raised the question why OWL is not a good candidate for encoding wordnets. On this issue, we agree with the assessment of Soria et al. (2009) who point out that *"[…] RDF and OWL are conceptual repositories representation formats that are not designed to represent polysemy and store linguistic properties of words and word meanings."*

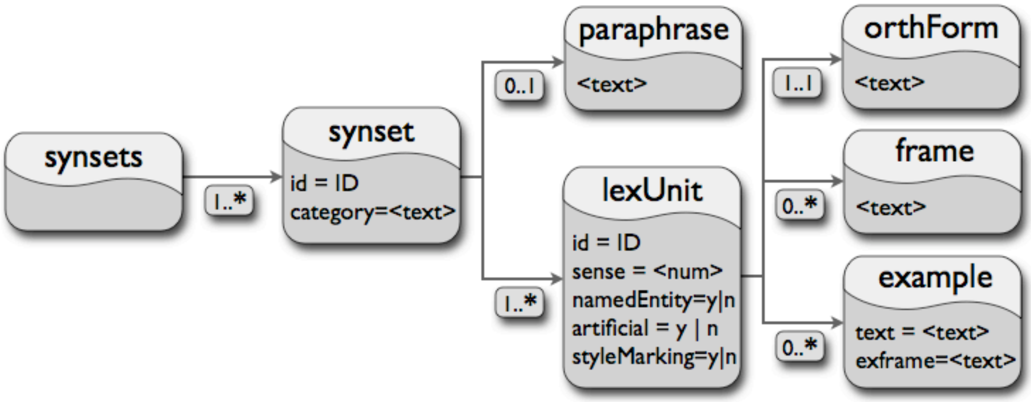


Figure 1. Structure of the XML synset files.

3 Current GermaNet XML Format

The structure of the XML files closely follows the internal structure of GermaNet, which means that the file structure mirrors the underlying relational organization of the data. There are two DTDs that jointly describe the XML-encoded GermaNet. One DTD represents all synsets with their lexical units and their attributes (see subsection 3.1). The other DTD represents all relations, both conceptual and lexical relations (see subsection 3.2).

The GermaNet XML format was initially developed by Kunze and Lemnitzer (2002), but modifications of the GermaNet data itself led to an adopted XML format, which is presented here.[8]

3.1 XML Synset Files

The XML files that represent all synsets and lexical units of GermaNet are organized around the three word categories currently included in GermaNet: nouns, adjectives, and verbs (altogether 54 synset files since the semantic space for each word category is divided into a number of semantic subfields).

The structure of each of these files is illustrated in Figure 1[9]. Each *synset* represents a set of lexical units (*lexUnits*) which all express the same meaning. This grouping represents the semantic relation of synonymy. Further properties of a *synset* (e.g., the word *category* or a describing *paraphrase*) and a lexical unit (e.g., a *sense* number or the orthographical form (*orthForm*)) are encoded appropriately.

Figure 1 describes the underlying XML structure. Each box in the figure stands for an element in the XML files, and the properties in each box (listed underneath the wavy line) represent the attributes of an XML element. This means, for example, that a *synset* element has the attributes of an *id* and a *category*.[10]

Figure 2 shows an example of a *synset* with two lexical units (*lexUnit* elements) and a *paraphrase*. The *lexUnit* elements in turn contain several attributes and an orthographical form (the *orthForm* element), e.g., *leuchten* (German verb for: *to shine*). The first of the two lexical units even has a *frame* and an *example*.

```
<synset id="s58377" category="verben">
  <lexUnit id="l82207"
           sense="1"
           namedEntity="no"
           artificial="no"
           styleMarking="no">
    <orthForm>leuchten</orthForm>
    <frame>NN</frame>
    <example>
      <text>
        Der Mond leuchtete in der Nacht.
      </text>
      <exframe>NN</exframe>
    </example>
  </lexUnit>
  <lexUnit id="l82208"
```

[8] The interested reader might compare the version at hand with (Lemnitzer and Kunze, 2002) or (Kunze and Lemnitzer, 2002), which both describe the initial GermaNet XML version.

[9] In fact, this figure is not quite complete for the reason of simplicity.

[10] Note that XML element or attribute names appear *italic* if they are referenced in the text.

```
            sense="2"
            namedEntity="no"
            artificial="no"
            styleMarking="no">
      <orthForm>strahlen</orthForm>
   </lexUnit>
   <paraphrase>
      Lichtstrahlen aussenden,
      große Helligkeit verbreiten
   </paraphrase>
</synset>
```

Figure 2. Synset file example.

3.2 XML Relation File

This type of XML file represents both kinds of relations: conceptual and lexical relations. All relations are encoded within one XML file, whose structure is illustrated in Figure 3.

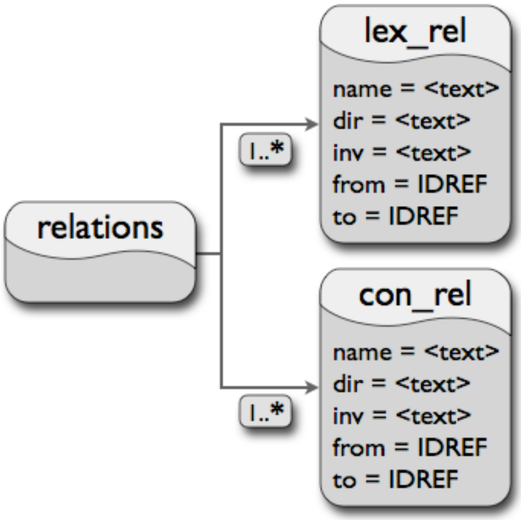


Figure 3. Structure of the XML relation file.

The boxes in Figure 3 again represent XML elements, which means that there is one *relations* element that contains all lexical relations (*lex_rel* elements) and conceptual relations (*con_rel* elements). Both relation types contain several attributes.

Figure 4 illustrates an example for each of the two relation types. The type of the conceptual relation is *hyperonymy* (indicated by the *name* attribute), and it holds between the synset with ID *s58377* (*from* attribute) and the synset with ID *s58376* (*to* attribute). The lexical relation is of type *antonymy* (again indicated by the *name* attribute), and holds between the lexical units with the IDs *l2471* (*from* attribute) and *l12470* (*to* attribute).

```
<con_rel name="hyperonymy"
         from="s58377" to="s58376"
         dir="revert" inv="hyponymy" />
<lex_rel name="antonymy"
         from="l2471" to="l2470"
         dir="both" />
```

Figure 4. Example from relation file.

4 Wordnet-LMF

The Lexical Markup Framework (ISO-24613) is an ISO standard for encoding natural language processing lexicons and machine readable dictionaries (Francopoulo et al., 2006). The intention of LMF is to provide a common model for the creation and use of lexical resources, to manage the exchange of data between and among these resources, and to enable the merging of a large number of individual electronic resources to form extensive global electronic resources.

4.1 The Challenge

The core structure of LMF is based on the prototypical structuring of a lexicon in terms of lexical entries, each of which enumerates the different senses of the lexical item in question. This word-driven perspective contrasts the synset-driven relational structure of wordnets – the grouping of word senses (i.e., lexical units) that express the same meaning into synsets. Exactly these two radically different organizing principles (relation-based in the case of wordnets versus lexical-entry-based in the case of LMF) constitute the challenge of encoding wordnets in LMF. We take up this challenge: *How can a synset-based wordnet, e.g. GermaNet, be represented in a word-driven format like LMF?*

4.2 Apply LMF to Wordnets

The conversion of GermaNet to LMF will build on Wordnet-LMF (Soria et al., 2009; Lee et al., 2009), an existing Lexical Markup Framework subset[11]. Wordnet-LMF has been developed in the context of the EU KYOTO

[11] Wordnet-LMF is a proper subset of LMF since there are specifications in LMF that are not in Wordnet-LMF and since there is nothing in Wordnet-LMF which is not in LMF. Soria et al. (2009) themselves refer to Wordnet-LMF as an LMF *dialect*.

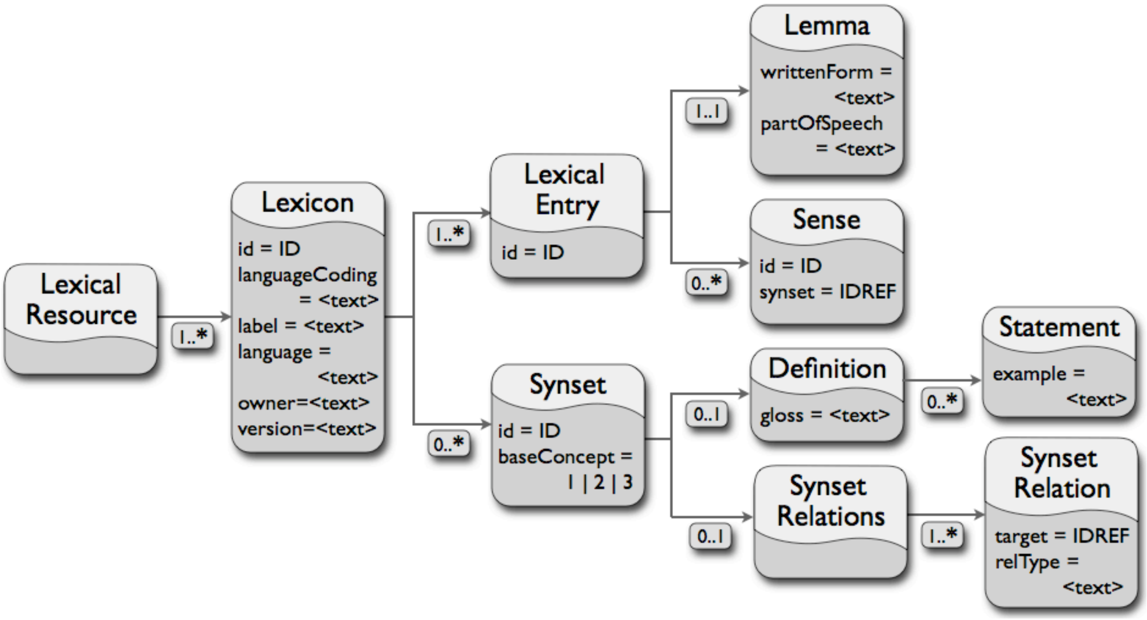


Figure 5. The Wordnet-LMF structure.

project[12] and is especially tailored to encode wordnets in the LMF standard.

Wordnet-LMF is specified by a Document Type Definition (see Appendix E in (Soria and Monachini, 2008)) and fully complies with standard LMF.

The Wordnet-LMF XML structure is shown in Figure 5[13]. There is a *Lexical Resource* which contains at least one *Lexicon* (in this case a wordnet lexicon).[14] A *Lexical Entry* represents a word entry in a *Lexicon*, where the word itself is represented by the *writtenForm* attribute of the *Lemma* element. *Lexical Entries* group different *Senses* of a particular word. The *Senses* have a *synset* attribute that relates them to a *Synset* element by the corresponding ID. If two *Senses* have the same *synset* attribute, they belong to the same *Synset* and are thus synonyms.

A *Synset* can have several relations to other *Synsets*. These relations are encoded in *SynsetRelation* elements.

[12] See http://www.kyoto-project.eu

[13] Note that this figure does not show the whole Wordnet-LMF model. Only the monolingual part that is relevant for this paper is represented. The representation of multilingual resources (i.e., the optional *SenseAxis* element with its children) is not considered in this paper. For a complete picture, see Soria et Monachini (2008).

[14] Here, XML element or attribute names again appear *italic* if they are referenced in the text.

4.3 Apply Wordnet-LMF to GermaNet

The differences between the synset-driven structure of GermaNet (see Figures 1 and 3) and the word-driven format of Wordnet-LMF (see Figure 5) are obvious. But there is also a strong commonality: Both formats have synset elements that cluster synonymous words. In GermaNet, the words are represented by lexical units that are child elements of a synset. In Wordnet-LMF, senses, which correspond to the lexical units in GermaNet, are linked to a synset (by an attribute containing a synset ID).

The conversion of GermaNet to Wordnet-LMF proceeds as follows: Each lexical unit of GermaNet is turned into a *Sense* element in Wordnet-LMF (see Figure 5). The *synset* attribute (containing a *Synset* ID) of the *Sense* element links this *Sense* with the *Synset* that it is a member of. The different *Sense* elements are grouped by their orthographical form (the *Lemma* in Wordnet-LMF) into *Lexical Entries*.

An example of a GermaNet *LexicalEntry* in Wordnet-LMF is shown in Figure 6. This *LexicalEntry* represents the word *leuchten* (German verb for: *to shine*), as the *writtenForm* attribute of the *Lemma* element indicates. This *LexicalEntry* has two *Senses*, which belong to different *Synsets* (see the different *synset* attributes of the *Sense* elements).

Each *Sense* has a *MonolingualExternalRefs* element with at least one *MonolingualExternalRef* representing a reference to an external system. In this case, each *Sense* is linked to the corresponding entry in the GermaNet database[15]; the *externalReference* attribute of a *MonolingualExternalRef* specifies the database table name with a database ID.

```
<LexicalEntry id="deu-52-l4601-v">
  <Lemma writtenForm="leuchten"
                      partOfSpeech="v" />
  <Sense id="deu-52-l4601-v_1"
               synset="deu-52-s58377-v">
    <MonolingualExternalRefs>
      <MonolingualExternalRef
        externalSystem="GermaNet-Database"
        externalReference=
               "lex_uni_table#id=82207" />
    </MonolingualExternalRefs>
  </Sense>
  <Sense id="deu-52-l4601-v_2"
               synset="deu-52-s58718-v">
    <MonolingualExternalRefs>
      <MonolingualExternalRef
        externalSystem="GermaNet-Database"
        externalReference=
               "lex_uni_table#id=82677" />
    </MonolingualExternalRefs>
  </Sense>
</LexicalEntry>
```

Figure 6. Example of a *LexicalEntry*.

In the next conversion step, all synsets of GermaNet are listed with their relations to other synsets. The corresponding *Synset* (with the ID *deu-52-s58377-v*) of the first *Sense* in Figure 6 is illustrated in Figure 7. It has, inter alia, a describing *gloss* and two *example* sentences.

The element *SynsetRelations* encodes relations to other *Synset* instances. The relations are simply encoded with a *target* attribute that contains the ID of the referencing *Synset*. The *Synsets* in Wordnet-LMF are logically the "same" as the synsets in GermaNet XML, i.e. the concept that a synset expresses is exactly the same in both formats.

Each *Synset* has a reference to the GermaNet database. Therefore, the *MonolingualExternalRef* element links to the corresponding entry in the GermaNet database; the *externalReference* attribute specifies the database table name with the synsets database ID.

```
<Synset id="deu-52-s58377-v"
                     baseConcept="1">
  <Definition gloss="Lichtstrahlen
             aussenden, große Helligkeit
             verbreiten">
    <Statement example="Der Mond leuchtete
                         in der Nacht."/>
    <Statement example="Die Lichter der
           Stadt strahlen in die Nacht."/>
  </Definition>
  <SynsetRelations>
    <SynsetRelation
              target="deu-52-s58376-v"
              relType="has_hyperonym"/>
  </SynsetRelations>
  <MonolingualExternalRefs>
    <MonolingualExternalRef
         externalSystem="GermaNet-Database"
         externalReference=
              "synset_table#id=58377"/>
  </MonolingualExternalRefs>
</Synset>
```

Figure 7. Example of a *Synset*.

These two Figures 6 and 7 represent the same example in Wordnet-LMF that was already shown in the GermaNet XML format in Figure 1.

4.4 Necessary Modifications to Wordnet-LMF

As the previous discussion has shown, Wordnet-LMF provides a very useful basis for converting GermaNet into LMF. However, a number of modifications to Wordnet-LMF are needed if this conversion is to preserve all information present in the original resource. The present section will discuss a number of modifications to Wordnet-LMF that are needed for conversion of wordnets in general. In addition, we will also discuss a set of extensions to Wordnet-LMF that are needed for conversion of GermaNet in particular.

The most glaring omission in Wordnet-LMF concerns the modeling of lexical relations which hold between lexical units (i.e., *Senses* in the terminology of Wordnet-LMF). In the current Wordnet-LMF DTD only conceptual relations (i.e., *SynsetRelations* in the terminology of Wordnet-LMF), which hold between synsets, are modeled. Thus antonymy, which is a typical example of a lexical relation (see (Fellbaum, 1998) for further details), can cur-

[15] For efficency reasons, GermaNet is stored in a relational database.

rently not be modeled without violating the Wordnet-LMF DTD.

Among the synset relations specified in Wordnet-LMF, the entailment relation is missing, which plays a crucial role in the modeling of verbs in the Princeton WordNet and in GermaNet alike. The list of values of attribute *relType* for *SynsetRelation* elements (see Appendix A in (Soria and Monachini, 2008)) therefore has to be amended accordingly.[16]

A third omission in the current Wordnet-LMF DTD concerns syntactic frames used in the Princeton WordNet to indicate the syntactic valence of a given word sense. Syntactic frames are also used in GermaNet, albeit using a different encoding[17]. Syntactic frames together with example sentences, which illustrate the meaning and prototypical usage of a particular word, help to distinguish among word senses.

In WordNet both syntactic frames and examples are linked to synsets. However, at least in the case of syntactic frames the linkage to synsets seems problematic since different members of the same synset may well have different valence frames. For example, the German verbs *finden* and *begegnen* both mean *meet* and thus belong to the same synset. Both are transitive verbs, but their object NPs have different cases: accusative case for *treffen* and dative case for *begegnen*. As this example shows, syntactic frames need to be associated with lexical units rather than synsets. This is exactly the design choice made in GermaNet, as shown in Figure 1.

A related question concerns the anchoring of example sentences which illustrate the meanings and prototypical usage of a particular word sense. In both the Princeton WordNet and GermaNet such examples are associated with lexical units[18]. GermaNet correlates examples additionally with particular syntactic frames and treats both examples and syntactic frames as properties of lexical units, i.e. *Senses* in the terminology of Wordnet-LMF.

The above issues lead to a modified version of the Wordnet-LMF DTD as shown in Figure 8. Compared to Figure 5, the *Sense* element is enriched by three optional subelements: *SenseRelations*, *SenseExamples*, and *SubcategorizationFrames*.

It has to be noted, though, that LMF proper contains all necessary elements. The three notions *SenseRelation*, *SenseExample*, and *SubcategorizationFrame* come from LMF proper and these elements can be used to remedy the omissions in Wordnet-LMF.

The *SenseRelation* element in Figure 8 represents relations between different *Senses* (the lexical units in GermaNet). The *SenseExamples* and *SubcategorizationFrames* elements both group several *SenseExample* or *SubcategorizationFrame* instances. A *SubcategorizationFrame* element represents the syntactic valence of a word sense. A *SenseExample* shows the prototypical usage of a word sense as an example sentence. The syntactic valence for a concrete example sentence can be specified with the optional *frame* attribute of a *SenseExample*.

5 Conclusion: Comparing GermaNet XML with Wordnet-LMF XML

We would like to conclude with a comparison between the GermaNet native XML format described in section 3 and the modified Wordnet-LMF format described in section 4.4. Since the GermaNet native XML format was particularly tailored to the structure of GermaNet, it enjoys the usual advantages of such customized solutions: it contains all and only the necessary XML elements and attributes to describe the resource. Moreover, the data are distributed over 55 different XML files, which facilitates easy data handling and efficient search by word classes and lexical fields. These properties are in fact exploited by a number of GermaNet-specific tools, including

[16] Piek Vossen (personal communication) has pointed out to us that Wordnet-LMF does not impose a list of relations as a standard yet.

[17] In WordNet, frames are encoded in a controlled language using paraphrases such as *Somebody ----s something* for a transitive verb with an animate subject and an inanimate object. The frames in GermaNet use complementation codes provided with the German version of the CELEX Lexical Database (Baayen et al., 2005) such as *NN.AN* for transitive verbs with accusative objects.

[18] In WordNet, the examples are placed at the synset level, but referencing to a word sense at the same time.

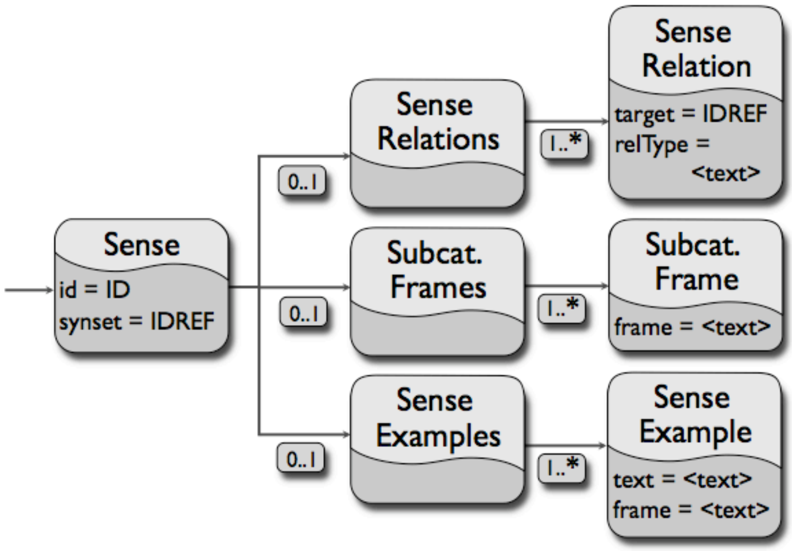


Figure 8. Revised Wordnet-LMF structure.

a GermaNet-Explorer, a tool for data exploration and retrieval, and a GermaNet Pathfinder, a tool for the calculation of semantic relatedness, similarity, and distance (Cramer and Finthammer, 2008). All of these tools utilize the Java API that has been developed for the GermaNet native XML format.

At the same time the GermaNet native XML format is a proprietary data format that was developed at a time when the only de facto encoding standard for wordnets consisted of the lexicographer files, originally developed for the Princeton WordNet. As such GermaNet XML was never developed with the goal of providing an XML standard for modeling wordnets in general. With Wordnet-LMF a candidate standard has now been proposed that is compliant with the LMF ISO standard for lexical resources and that strives to provide a general encoding standard of wordnets for different languages. As the discussion in section 4.4 has shown, the current Wordnet-LMF DTD still needs to be amended to account for the full range of wordnet relations, frames, and examples (see Figure 8). These elements are not in Wordnet-LMF because Wordnet-LMF is a subset, but these elements are defined in the ISO document 24613 where LMF proper is defined. However, Wordnet-LMF appears to be suitably mature to serve as an interchange format for wordnets of different languages as well as for linking wordnets of different languages with one another[19].

Acknowledgements

The research reported in this paper was funded by the BW-eSci(T) grant sponsored by the Ministerium für Wissenschaft, Forschung und Kunst Baden-Württemberg.

We would like to thank Piek Vossen and an anonymous reviewer for valuable comments on an earlier version of this paper.

References

Baayen, R. H., R. Piepenbrock, and L. Gulikers. 2005. *The CELEX Lexical Database (Release 2) CD-ROM.* Philadelphia, PA: Linguistic Data Consortium, University of Pennsylvania (Distributor).

Cramer, Irene, and Marc Finthammer. 2008. Tools for Exploring GermaNet in the Context of CL-Teaching. In: *Angelika Storrer, Alexander Geyken, Alexander Siebert, and Kay-Michael Würzner, (Eds.): Text Resources and Lexical Knowledge*. Selected Papers from the 9th Conference on Natural Language Processing KONVENS 2008. Berlin/New York: Mouton de Gruyter, 195-208.

[19] For example, the Interlingual Index, based on the Princeton WordNet, can be used to link different wordnets with one another.

Kunze, Claudia, and Lothar Lemnitzer. 2002. GermaNet – representation, visualization, application. *Proceedings of LREC 2002*, main conference, Vol V. pp. 1485-1491.

Fellbaum, Christiane (eds.). 1998. *WordNet – An Electronic Lexical Database*. The MIT Press.

Francopoulo, Gil, Monte George, Nicoletta Calzolari, Monica Monachini, Nuria Bel, Mandy Pet, and Claudia Soria. 2006. Lexical markup framework (LMF). *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*. Genoa, Italy.

Henrich, Verena, and Erhard Hinrichs. 2010. GernEdiT – The GermaNet Editing Tool. *Proceedings of LREC 2010*, main conference. Valletta, Malta.

Lee, Lung-Hao, Shu-Kai Hsieh, and Chu-Ren Huang. 2009. CWN-LMF: Chinese WordNet in the Lexical Markup Framework. *Proceedings of the 7th Workshop on Asian Resources*. Suntec, Singapore, August 06 - 07, 2009, pp. 123-130

Lemnitzer, Lothar, and Claudia Kunze. 2002. Adapting GermaNet for the Web. *Proceedings of the First Global Wordnet Conference*. Central Institute of Indian Languages, Mysore, India, 21.-25.01.2002, pp. 174-181

Soria, Claudia, Monica Monachini, and Piek Vossen. 2009. Wordnet-LMF: Fleshing out a Standardized Format for Wordnet Interoperability. *Proceedings of ACM Workshop on Intercultural Collaboration*.

Soria, Claudia, and Monica Monachini. 2008. *Kyoto-LMF – Wordnet representation format*. KYOTO Working paper: WP02_TR002_V04_Kyoto_LMF.

Vossen, Piek, Eneko Agirre, Nicoletta Calzolari, Christiane Fellbaum, Shu-kai Hsieh, Chu-Ren Huang, Hitoshi Isahara, Kyoko Kanzaki, Andrea Marchetti, Monica Monachini, Federico Neri, Remo Raffaelli, German Rigau, Maurizio Tescon, and Joop VanGent. 2008. KYOTO: A system for mining, structuring and distributing knowledge across languages and cultures. *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*. Marrakech, Morocco.

Normal-form parsing for Combinatory Categorial Grammars with generalized composition and type-raising

Julia Hockenmaier **Yonatan Bisk**
University of Illinois at Urbana-Champaign
{juliahmr, bisk1}@illinois.edu

Abstract

We propose and implement a modification of the Eisner (1996) normal form to account for generalized composition of bounded degree, and an extension to deal with grammatical type-raising.

1 Introduction

Combinatory Categorial Grammar (Steedman, 2000) is a linguistically expressive grammar formalism that has been used for many NLP applications, including wide-coverage parsing (Clark and Curran, 2007; Hockenmaier, 2003) and semantic interpretation (Curran et al., 2007), semantic role-labeling (Gildea and Hockenmaier, 2003; Boxwell et al., 2009), semantic parsing (Zettlemoyer and Collins, 2005) and natural language generation (Espinosa et al., 2008).

An essential feature of CCG is its flexible constituent structure, licensed by type-raising and composition rules which can create "non-standard" constituents such as *"John saw"*, or *"Mary talked to"*, required in constructions involving non-local dependencies, such as wh-extraction (Fig. 1) or right-node raising. Since *"John saw"* can now also be a constituent in *"John saw Mary"*, this leads to a combinatorial explosion of *spurious ambiguities*, i.e. multiple syntactic derivations of the same semantic interpretation (Wittenburg, 1986). This can create problems for applications based on CCG, e.g. for the induction of stochastic CCGs from text annotated with logical forms (Zettlemoyer and Collins, 2007), where spreading probability mass over equivalent derivations should be avoided. A number of *normal-form* (NF) parsing algorithms that aim to produce only one derivation per interpretation have been proposed (Wittenburg, 1986; Niv, 1994; Pareschi and Steedman, 1987; Hepple and Morrill, 1989; Eisner, 1996). Computationally, such algorithms are very attractive since they do not require costly semantic equivalence checks (Karttunen, 1989; Komagata, 2004) during parsing. Eisner's (1996) normal form is the most developed and well-known of these approaches, but is only defined for a variant of CCG where type-raising is a lexical operation and where the degree of composition is unbounded. Therefore, it and its equivalent reformulation by Hoyt and Baldridge (2008) in a multimodal variant of CCG are not safe (preserve all interpretations) and complete (remove all spurious ambiguities) for more commonly used variants of CCG. In particular, this NF is not safe when the degree of composition is bounded,[1] and not complete when type-raising is a grammatical operation. This paper defines a NF for CCG with bounded composition and grammatical type-raising.

2 Combinatory Categorial Grammar

In CCG, every constituent (*"John saw"*) has a syntactic category (S/NP) and a semantic interpretation ($\lambda x.saw(john', x)$).[2] Constituents combine according to a small set of language-

[1] Although Eisner (1996, section 5) also provides a safe and complete parsing algorithm which can return non-NF derivations when necessary to preseve an interpretation if composition is bounded or the grammar is restricted in other (arbitrary) ways.

[2] More complex representations than simple predicate-argument structures are equally possible.

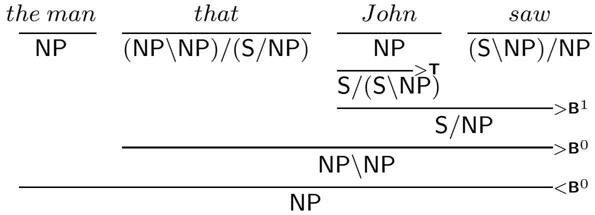


Figure 1: CCG derivations for wh-extraction

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 465–473, Beijing, August 2010

Application	($>$)	$\mathsf{X/Y}: \lambda x.f(x)$	$\mathsf{Y}: a$	$\Rightarrow \mathsf{X}: f(a)$
	($<$)	$\mathsf{Y}: a$	$\mathsf{X\backslash Y}: \lambda x.f(x)$	$\Rightarrow \mathsf{X}: f(a)$
Composition	($>\mathbf{B}^1$)	$\mathsf{X/Y}: \lambda x.f(x)$	$\mathsf{Y/Z}: \lambda y.g(y)$	$\Rightarrow \mathsf{X/Y}: \lambda z.f(g(z))$
	($<\mathbf{B}^1$)	$\mathsf{Y\backslash Z}: \lambda y.g(y)$	$\mathsf{X\backslash Y}: \lambda x.f(x)$	$\Rightarrow \mathsf{X\backslash Y}: \lambda z.f(g(z))$
	($>\mathbf{B}^1_\times$)	$\mathsf{X/Y}: \lambda x.f(x)$	$\mathsf{Y\backslash Z}: \lambda y.g(y)$	$\Rightarrow \mathsf{X\backslash Y}: \lambda z.f(g(z))$
	($<\mathbf{B}^1_\times$)	$\mathsf{Y/Z}: \lambda y.g(y)$	$\mathsf{X\backslash Y}: \lambda x.f(x)$	$\Rightarrow \mathsf{X/Y}: \lambda z.f(g(z))$
	($>\mathbf{B}^n$)	$\mathsf{X/Y}: \lambda x.f(x)$	$\mathsf{Y\|Z_1\|...\|Z_n}: \lambda z_n..z_1.g(z_1...z_n)$	$\Rightarrow \mathsf{X\|Z_1\|...\|Z_n}: \lambda z_n...z_1.f(g(z_1...z_n))$
	($<\mathbf{B}^n$)	$\mathsf{Y\|Z_1\|...\|Z_n}: \lambda z_n..z_1.g(z_1...z_n)$	$\mathsf{X\backslash Y}: \lambda x.f(x)$	$\Rightarrow \mathsf{X\|Z_1\|...\|Z_n}: \lambda z_n...z_1.f(g(z_1...z_n))$
Typeraising	($>\mathbf{T}$)	For $\mathsf{X} \in \mathcal{C}_{arg}$: $\mathsf{X}: a$		$\Rightarrow \mathsf{T/_i(T\backslash_i X)}: \lambda f.f(a)$
	($<\mathbf{T}$)	For $\mathsf{X} \in \mathcal{C}_{arg}$: $\mathsf{X}: a$		$\Rightarrow \mathsf{T\backslash_i(T/_i X)}: \lambda f.f(a)$

Figure 2: CCG's combinatory rules.

independent combinatory rules (Fig. 2). The lexicon pairs words with categories and interpretations and is language-specific.

Syntax We distinguish atomic (S, NP, PP, etc.) from complex categories ((S\NP)/NP, N/N, etc.). A complex category of the form X/Y (or X\Y) represents a function which returns a *result* of type X when applied to an *argument* of type Y, which, in the case of a forward slash (/) has to follow the functor, and in the case of a backslash (\) has to preceed it. X and Y can themselves be complex again. We will use categories with vertical slashes when the direction of the slash does not matter, and may omit unnecessary parentheses (so X|Y|Z will represent (X\Y)/Z, (X\Y)\Z, ...). We will also use the shorthand $\mathsf{X|Y_{1..n}}$ (or $\mathsf{X}|\alpha$) to refer to a category with (possibly complex) result X and arguments $\mathsf{Y_1...Y_n}$ (or an unspecified, possibly empty, list of arguments $\alpha = Y_{0...n}$, where $|\alpha| = n$) that can each appear with either type of slash.

Semantics If the category of a constituent is atomic (NP; S), its interpretation will also be atomic (*kim'*; *sleeps'(kim')*), and if the category is a functor of arity n ($\mathsf{X|Y_{1..n}}$), the interpretation is a λ-expression $\lambda y_n..\lambda y_1 \phi(y_1...y_n)$ of arity n.

The lexicon Each language defines a finite set of lexical category types $\mathcal{C}_{lex}$ (e.g. (S\NP)/NP is in the English lexicon, but (S\NP)\NP is not) with maximal arity N_L. This defines a set of lexical argument category types $\mathcal{C}_{arg}$, consisting of all categories Y that are the argument of some lexical category $(\mathsf{X|Y})|\beta \in \mathcal{C}_{lex}$ (with $|\beta| \geq 0$). Since $\mathcal{C}_{lex}$ is finite, $\mathcal{C}_{arg}$ is strictly smaller than $\mathcal{C}_{lex}$ (and usually consists of basic categories such as NP, S, S\NP).

Combinatory Rules In addition to function application ($>, <$), CCG has three kinds of combinatory rules (Fig. 2): harmonic function composition ($>\mathbf{B}^{(1)}$, $<\mathbf{B}^{(1)}$), crossing function composition ($>\mathbf{B}_\times$,$<\mathbf{B}_\times$) and type-raising ($>\mathbf{T}$, $<\mathbf{T}$). All rules take one or two *input* categories and yield one *output* category, and consist of a syntactic and a corresponding semantic operation. Composition also has *generalized* variants $>\mathbf{B}^n$, $<\mathbf{B}^n$ up to a fixed degree $N_\mathbf{B}$.[3] Composition of unbounded degree increases the generative capacity of CCG (Weir, 1988), and should be disallowed. Application ($>, <$) can be seen as a special case of composition ($>\mathbf{B}^0$,$<\mathbf{B}^0$). When composing X|Y with Y|Z to X|Z, we call X|Y the *primary* input and Y|Z the *secondary* input. Harmonic composition allows associativity: the string A/B B/C C now has an alternative derivation where A/B and B/C compose into A/C, whereas crossing composition enables novel permutations, such as C A/B B\C.

Type-raising swaps the functor-argument relation. Although it is often assumed to take place in the lexicon, we will distinguish lexical categories (e.g. for quantifiers) that have the syntactic type of type-raised categories, but semantics that could not be obtained by type-raising a simple category from grammatically type-raised categories. We follow the common definition of CCG (Steedman, 2000) and allow only categories $\mathsf{X} \in \mathcal{C}_{arg}$ to be type-raised.[4] Instantia-

[3]In $\mathsf{X|Y_{1..n}}$ or $\mathsf{X}|\alpha$=$\mathsf{X|Y_{1...|\alpha|}}$, we do not assume the slash variable $| \in \{/, \backslash\}$ to be instantiated the same way for all $\mathsf{Y_i}$. We will therefore only distinguish between forward and backward generalized composition $\mathbf{B}^{n>1}$.

[4]We stipulate that it may be further necessary to only allow those argument categories to type-raise that are not used to project unbounded dependencies, such as S/NP in

tions of the variable T should also be restricted to categories of finite arity $N_{\mathbf{T}}$ in oder to prevent an increase in generative capacity (Hoffman, 1995; Komagata, 1997). We refer to the arity of T as the degree of any particular instantation of $\mathbf{T}$. We follow Steedman (2000) and assume $N_{\mathbf{T}} = N_{\mathbf{B}}$.

Coordination requires a ternary rule (Φ) which can be binarized (Φ>, Φ<) to simplify parsing:[5]

$$\begin{array}{llll} (\Phi) & \mathsf{X} & \mathsf{conj}\ \mathsf{X} & \Rightarrow \mathsf{X} \\ (\Phi>) & \mathsf{X} & \mathsf{X[conj]} & \Rightarrow \mathsf{X} \\ (\Phi<) & \mathsf{conj} & \mathsf{X} & \Rightarrow \mathsf{X[conj]} \end{array}$$

Uses of type-raising and composition In English, type-raising and composition are required for wh-extraction and right node raising of arguments as well as so-called argument cluster coordination. In other languages, they are needed for scrambling and cross-serial dependencies.

It is important to note that when type-raising is *required*, it always occurs in tandem with composition. Since type-raising an argument Y to $\mathsf{X}/(\mathsf{X}\backslash\mathsf{Y})$ and applying it to the functor $\mathsf{X}\backslash\mathsf{Y}$ is semantically equivalent to applying $\mathsf{X}\backslash\mathsf{Y}$ directly to Y, type-raising is never required when function application can be used instead. That is, in all cases, a type-raised argument must be composed with another constituent, usually the original functor (head). Only in argument-cluster coordination will the type-raised element be composed with a non-head constituent. In the latter case, coordination will be required before the argument cluster can be combined with the head. Composition without type-raising may occur, e.g. for adjuncts, which have categories $\mathsf{X}|\mathsf{X}$, but may modify a constituent with category $\mathsf{X}|\alpha$.

Restrictions on type-raising and composition In order to prevent overgenerations of the form *"John speaks because Chinese, he enjoys Beijing."*, we assume a variant of CCG in which forward crossing composition $>\mathbf{B}^{1}_{\times}$ (e.g. of *because*:$(\mathsf{S}/\mathsf{S})/\mathsf{S}$) into the result of backward type-raising $<\mathbf{T}$ (e.g. *Chinese*:$\mathsf{S}\backslash(\mathsf{S}/\mathsf{NP})$, and, similarly, $<\mathbf{B}^{x}$ into the result of $>\mathbf{T}$, are disallowed.

$(\mathsf{NP}\backslash\mathsf{NP})/(\mathsf{S}/\mathsf{NP})$ for English object relative pronouns.

[5]Here, X needs to be restricted to a finite set of categories (Weir, 1988). In multimodal CCG, conjunction have categories of the form $(\mathsf{X}_\star\backslash_\star\mathsf{X})/_\star\mathsf{X}$, i.e. must apply to their argument

Punctuation and Type-changing rules CCGbank (Hockenmaier and Steedman, 2007) uses special punctuation rules such as $\mathsf{S}\ .\Rightarrow\mathsf{S}$ or $,\mathsf{NP}\backslash\mathsf{NP}\Rightarrow\mathsf{NP}\backslash\mathsf{NP}$, and a small number of (non-recursive) type-changing rules (with idiosyncratic semantics) such as $\mathsf{N}\Rightarrow\mathsf{NP}$ (for determiner-less NPs) or $\mathsf{S[pss]}\backslash\mathsf{NP}\Rightarrow\mathsf{NP}\backslash\mathsf{NP}$ (for complex adjuncts, here passive VPs being used as NP postmodifiers):

$$\begin{array}{lllll} \text{Punctuation} & (>\mathbf{P}) & \mathsf{X}{:}\phi\ [.,;] & \Rightarrow \mathsf{X}{:}\phi \\ & (<\mathbf{P}) & [.,;]\ \mathsf{X}{:}\phi & \Rightarrow \mathsf{X}{:}\phi \\ \text{TypeChanging} & (\mathbf{TCR}) & \mathsf{X}{:}\phi & \Rightarrow \mathsf{Y}{:}\psi(\phi) \end{array}$$

CCG parsing CCG can be parsed with a bottom-up CKY-like algorithm (Shieber et al., 1995; Steedman, 2000), which differs from standard CKY in that it requires one (or two) unary completion steps in each cell to deal with type-raising (and type changing).[6] Chart items are of the form $\langle\mathsf{X}, i, j\rangle$, where X is a category, and the indices i and j represent the span of the item. Interpretations need only to be constructed for complete derivations when unpacking the chart.

3 The Eisner normal form

The Eisner normal form Eisner (1996) presents a normal-form parsing algorithm for CCG without grammatical type raising (where the lexicon may still contain categories like $\mathsf{S}/(\mathsf{S}\backslash\mathsf{NP})$, but there is no combinatory rule that changes a complex (derived) NP to e.g. $\mathsf{S}/(\mathsf{S}\backslash\mathsf{NP})$). He proves that his algorithm finds only one canonical derivation for each semantic interpretation of an input string consisting of a sequence of words and their lexical categories. Since the presence of both pre- and postmodifiers (as in *"intentionally knock twice"*[7]) introduces a genuine ambiguity, Eisner proves that the only kind of spurious ambiguity that can arise in his variant of CCG is due to associative chains of composition such as $\mathsf{A}/\mathsf{B}\ \mathsf{B}/\mathsf{C}\ \mathsf{C}/\mathsf{D}$ or $\mathsf{A}/\mathsf{B}\ \mathsf{B}/\mathsf{C}\ \mathsf{C}\backslash\mathsf{D}$, which can be derived as either

[6]Since composition allows the arity of derived ($\approx$ non-terminal) CCG categories to grow with the length of the input string, worst-case complexity of this naive algorithm is exponential. (Vijay-Shanker and Weir, 1993)'s $O(n^6)$ algorithm has a more compact representation of categories.

[7]This can mean $\lambda x.intentionally'(twice'(knock'(x)))$ or $\lambda x.twice'(intentionally'(knock'(x)))$.

Eisner NF	Not Eisner NF
$(A\|B_{1..b})/C \quad (C\|D_{1..d})/E \quad (E\|F_{1..f})/G \quad G\|H_{1..h}$	$(A\|B_{1..b})/C \quad (C\|D_{1..d})/E \quad (E\|F_{1..f})/G \quad G\|H_{1..h}$
$(E\|F_{1..f})/G \; G\|H_{1..h} \Rightarrow_{>\mathbf{B}^h} (E\|F_{1..f})\|H_{1..h}$	$(A\|B_{1..b})/C \; (C\|D_{1..d})/E \Rightarrow_{>\mathbf{B}^{d+1}} ((A\|B_{1..b})\|D_{1..d})/E$
$\Rightarrow_{>\mathbf{B}^{f+h}} ((C\|D_{1..d})\|F_{1..f})\|H_{1..h}$	$\Rightarrow_{>\mathbf{B}^{f+1}} (((A\|B_{1..b})\|D_{1..d})\|F_{1..f})\|G$
$\Rightarrow_{>\mathbf{B}^{d+f+h}} (((A\|B_{1..b})\|D_{1..d})\|F_{1..f})\|H_{1..h}$	$\Rightarrow_{>\mathbf{B}^{h}} (((A\|B_{1..b})\|D_{1..d})\|F_{1..f})\|H_{1..h}$

Figure 3: Eisner NF and generalized composition $\mathbf{B}^{n>1}$

Left branching		Right branching		
$>\mathbf{B}^0(>\mathbf{B}^{m+1},...)$	$\Rightarrow$	$>\mathbf{B}^{m\geq 0}(...,>\mathbf{B}^0)$	A/B $(B\|D_{0..m})$/C C	$m \geq 0$
$>\mathbf{B}^1(>\mathbf{B}^{m\geq 1},...)$	$\Rightarrow$	$>\mathbf{B}^{m\geq 1}(...,>\mathbf{B}^1)$	A/B $(B\|C_{1...m-1})$/D D/E	$m \geq 1$
$>\mathbf{B}^{n\geq 1}(>\mathbf{B}^1,...)$	$\Rightarrow$	$>\mathbf{B}^{n}(...,>\mathbf{B}^{m=n})$	A/B B/C $C/D_{1..n}$	$m = n \geq 1$
$\emptyset$	$\not\Leftarrow$	$>\mathbf{B}^{n>1}(...,>\mathbf{B}^{m>n})$	$A/(B\|D_{1..k})$ B/C $((C\|D_{1..k})\|E_{1..n}$	$m > n > 1$
$>\mathbf{B}^{m}(>\mathbf{B}^k,...)$	$\Leftarrow$	$>\mathbf{B}^{n>1}(...,>\mathbf{B}^{1<m<n})$	A/B $(B\|C_{1..k-1})$/D $D\|E_{1..m}$	$n > m > 1$

Figure 4: Associative composition chains: our NF disallows the grayed-out derivations.

$>\mathbf{B}$ (..., $>\mathbf{B}$) or $>\mathbf{B}$ ($>\mathbf{B}$, _). This is eliminated by the following constraint:

Eisner NF Constraint 1. *The output* $X|\alpha$ *of forward composition* $>\mathbf{B}^{n>0}$ *cannot be the primary input to forward application or composition* $>\mathbf{B}^{m\geq 0}$. *The output of* $<\mathbf{B}^{n>0}$ *cannot be the primary input to* $<\mathbf{B}^{m\geq 0}$.

This can be implemented by a ternary feature $H_E \in \{>\mathbf{B}^n, <\mathbf{B}^n, \emptyset\}$ and chart items of the form $\langle X, H_E, i, j\rangle$ where $H_E = >\mathbf{B}^n$ (or $<\mathbf{B}^n$) if X was produced by the corresponding composition rule (for any $n > 0$) and $\emptyset$ otherwise.

4 A new normal form for CCG

4.1 Generalized composition

Eisner NF and generalized composition Unboundedly long sequences of generalized composition are required e.g. for Dutch verb clusters that give rise to cross-serial dependencies ($N_1...N_nV_1...V_n$ with N_i the argument of V_i). These can be obtained through standard bounded-degree compositions, but the Eisner NF produces a derivation that requires compositions of unbounded degree (Fig. 3). Although this is allowed in the variant of CCG Eisner considers, compositions of unbounded degree are usually disallowed because they increase the generative capacity of CCG (Weir, 1988). We stipulate that the NF of any derivation τ should not require composition rules of higher degree than τ itself. Note that the output of function application ($\mathbf{B}^0$) always has lower arity than its functor; the output of regular composition ($\mathbf{B}^1$) has the same arity as its primary functor, but the output of generalized composition ($\mathbf{B}^{n>1}$) has an arity that is $n-1$ higher than that of the primary functor. $\mathbf{B}^{n>1}$ therefore requires a different treatment.

Our reformulation of the Eisner NF Associative composition chains for constituents A B C can lead to spurious ambiguity if both a left-branching $>\mathbf{B}^{\mathsf{n}}(>\mathbf{B}^{\mathsf{m}}(\mathsf{A\ B})\ \mathsf{C})$ and a right-branching $>\mathbf{B}^{\mathsf{n}'}(\mathsf{A}\ >\mathbf{B}^{\mathsf{m}'}(\mathsf{B\ C}))$ are possible and lead to the same interpretation. Figure 4 illustrates all possible cases consisting of three constituents. In most cases, the right-branching (Eisner NF) derivation is to be preferred. For generalized composition $>\mathbf{B}^{n>1}$, $>\mathbf{B}^{m>1}$, left-branching $>\mathbf{B}^{n>1}(>\mathbf{B}^{m>1},...)$ is always allowed, but right-branching $>\mathbf{B}^{n}(...,>\mathbf{B}^{m})$ is only allowed if $m \geq n$.

NF Constraint 1 ($\mathbf{B}^0$ and $\mathbf{B}^{n\geq 1}$)**.** *The output of* $>\mathbf{B}^{n\geq 1}$ *(resp.* $<\mathbf{B}^{n\geq 1}$*) cannot be primary functor for* $>\mathbf{B}^{n\leq 1}$ *(resp.* $<\mathbf{B}^{n\leq 1}$*).*

NF Constraint 2 ($\mathbf{B}^1$ and $\mathbf{B}^{n\geq 1}$)**.** *The output of* $>\mathbf{B}^1$ *(resp.* $<\mathbf{B}^1$*) cannot be primary functor for* $>\mathbf{B}^{n\geq 1}$ *(resp.* $<\mathbf{B}^{n\geq 1}$*).*

NF Constraint 3 ($\mathbf{B}^{n>1}$ and $\mathbf{B}^{m>1}$)**.** *The output of* $>\mathbf{B}^m$ *(resp.* $<\mathbf{B}^m$*) cannot be secondary functor for* $>\mathbf{B}^{n>m}$ *(resp.* $<\mathbf{B}^{n>m}$*).*

4.2 Grammatical type-raising

Eisner NF and type-raising Figure 5 illustrates a spurious ambiguity arising through type-

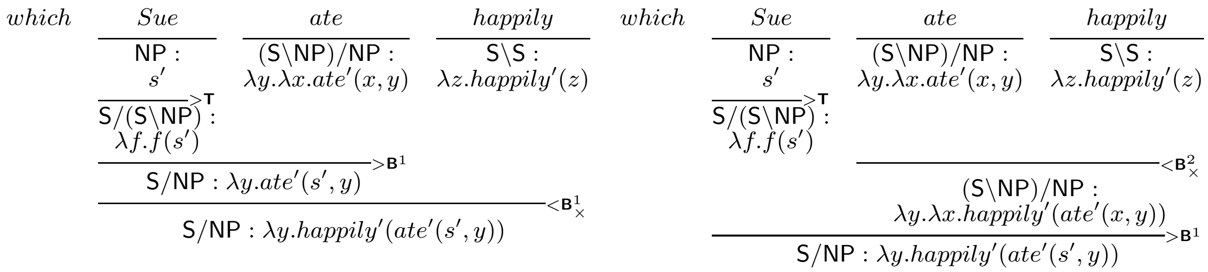
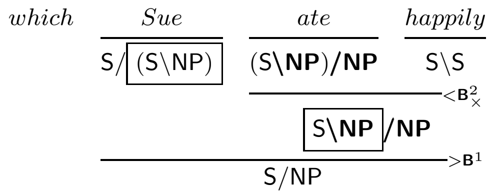


Figure 5: The Eisner NF allows spurious ambiguity arising due to type-raising

raising that the Eisner NF does not exclude.[8] Here two derivations can be obtained because the result of combining the adverb with the subject-verb cluster is no longer the output of a forward composition, and can therefore apply to the object. The derivations are semantically equivalent: although type-raising reverses the syntactic functor-argument relation, a type-raised argument applied to a predicate returns the same interpretation as when the predicate is applied directly to the original. But Eisner treats S/(S\NP) as a category with semantics $\lambda x.\phi(x)$, in which case the derivations yield indeed different scope relations. Eisner's analyis is correct for certain classes of words which have lexical categories that appear like type-raised categories, but have a different interpretation from that of categories obtained by type-raising. These are usually scope-bearing elements, such as the universal quantifer *every* ((S/(S\NP))/N : λPλQ∀xP(x) → Q(x)), and there may not be a single derivation which captures all semantic interpretations. Lexicalized pseudo-type-raising therefore needs to be distinguished from grammatical type-raising.

Our extension of the (modified) Eisner NF In Fig. 5, Eisner NF licenses two derivations. Both contain an instance of composition in which the type-raised argument is the primary component. In the analysis in which this is the second derivation step, the canceled part of this $<\mathbf{B}^2$ composition (boxed) contains a category (\NP) that was part of the argument output of the first $>\mathbf{B}^1$ composition (bold-faced):



Our NF will eliminate derivations of this type and prefer the other, lower-degree derivation. We stipulate that the spurious ambiguities that arise through type-raising and composition can be eliminated through the following rule:

NF Constraint 4 (**T** and $\mathbf{B}^{n>0}$). *The output of* $>\mathbf{T}$ *cannot be primary input to* $>\mathbf{B}^{n>0}$ *if the secondary input is the output of* $<\mathbf{B}^{m>n}$. *The output of* $<\mathbf{T}$ *cannot be primary input in* $<\mathbf{B}^{n>0}$ *if the secondary input is the output of* $>\mathbf{B}^{m>n}$.

We also stipulate that a type-raised T/(T\X) cannot be used as a functor in application (since T\X could always apply directly to X).

NF Constraint 5 (**T** and $\mathbf{B}^0$). *The output of forward (or backward) type-raising* $>\mathbf{T}$ *(resp.* $<\mathbf{T}$ *) cannot be the functor in application* $>$ *(resp.* $<$*).*

Additional spurious ambiguities arise through the interaction of type-raising and coordination: Since any category can be coordinated, we can either coordinate X and then type-raise the coordinated X to T/(T\X), or we can first type-raise each conjunct to T/(T\X) and then conjoin. Since nonsymmetric coordinations of an argument-adjunct cluster and a single argument (as in *eats ((pizza for lunch) and pasta)*) require type-raising before coordination, we formulate the following rule to eliminate interactions between type-raising and coordination:

NF Constraint 6 (**T** and Φ). *The result of coordination* Φ *cannot be type-raised.*

[8] We have chosen a slighly unusual adverb category to illustrate a general problem.

NF Derivation A

$$\dfrac{\dfrac{A}{\mathsf{X/X} : \lambda P a(P)} \quad \dfrac{\dfrac{B}{(\mathsf{X}|\alpha_\mathsf{a})|\beta_\mathsf{b} : \lambda \mathbf{x}_b \mathbf{x}_a b(\mathbf{x}_a \mathbf{x}_b)} \quad \dfrac{C}{(\mathsf{X}|\alpha_\mathsf{a})\backslash(\mathsf{X}|\alpha_\mathsf{a}) : \lambda Q \lambda \mathbf{z}_a c(Q(\mathbf{z}_a))}}{(\mathsf{X}|\alpha_\mathsf{a})|\beta_\mathsf{b} : \lambda \mathbf{x}_b \mathbf{x}_a c(b(\mathbf{x}_a \mathbf{x}_b))} {<}\mathbf{B}^b}{(\mathsf{X}|\alpha_\mathsf{a})|\beta_\mathsf{b} : \lambda \mathbf{x}_b \mathbf{x}_a a(c(b(\mathbf{x}_a \mathbf{x}_b)))} {>}\mathbf{B}_\times^{a+b}$$

NF Derivation B

$$\dfrac{\dfrac{\dfrac{A}{\mathsf{X/X} : \lambda P a(P)} \quad \dfrac{B}{(\mathsf{X}|\alpha_\mathsf{a})|\beta_\mathsf{b} : \lambda \mathbf{x}_b \mathbf{x}_a b(\mathbf{x}_a \mathbf{x}_b)}}{(\mathsf{X}|\alpha_\mathsf{a})|\beta_\mathsf{b} : \lambda \mathbf{x}_b \mathbf{x}_a a(b(\mathbf{x}_a \mathbf{x}_b))} {>}\mathbf{B}_\times^{a+b} \quad \dfrac{C}{(\mathsf{X}|\alpha_\mathsf{a})\backslash(\mathsf{X}|\alpha_\mathsf{a}) : \lambda Q \lambda \mathbf{z}_a c(Q(\mathbf{z}_a))}}{(\mathsf{X}|\alpha_\mathsf{a})|\beta_\mathsf{b} : \lambda \mathbf{x}_b \mathbf{x}_a c(a(b(\mathbf{x}_a \mathbf{x}_b)))} {<}\mathbf{B}_\times^b$$

Figure 6: Constituents with pre- and postmodifiers have two semantically distinct derivations

Punctuation and Type-changing rules Punctuation results in spurious ambiguities, either when a constituent X has both an initial and a final punctuation mark (e.g. a comma), or when it has an initial (final) punctuation mark and a final (initial) modifier. The first case is easy to fix by disallowing the output of , X $\Rightarrow$ X to be the input of X , $\Rightarrow$ X. The latter could be eliminated by disallowing the output X of right-recursive (left-recursive) punctuation rule to be secondary input to any left-recursive (right-recursive) application or composition rule (e.g. X X\X $\Rightarrow$ X).[9]

Implementation Our normal-form constraints can be implemented in a bottom-up parser with items of the form $\langle \mathsf{X}, \mathcal{C}, i, j\rangle$, with

$$\mathcal{C} \in \{>, {>}\mathbf{B}\,1, {>}\mathbf{B}\,2, ..., {>}\mathbf{B}^n; <, {<}\mathbf{B}\,1, {<}\mathbf{B}\,2, ..., {<}\mathbf{B}^n; {>}\mathbf{T}, {<}\mathbf{T}, {>}\mathsf{Pct}, {<}\mathsf{Pct}, \Phi{>}, \Phi{<}, \mathsf{TCR}\}$$

4.3 Is our normal form safe and complete?

Here we sketch the beginnings of a proof that our algorithm allows one and only one syntactic derivation per semantic interpretation for the version of CCG we consider. We first examine all cases of two adjacent constituents A, B which must combine into a category C:

Functor X/Y and argument Y combine to X The functor must apply to the argument. The argument could type-raise, but then cannot apply.

Functor X/Y|α and argument Y combine to X|α The functor cannot apply to the argument. The argument must type-raise to X\(X/Y), and can then backward-compose into the functor.

Functor X/X and X\X can combine to X/X or X\X This is not a spurious ambiguity, since the output categories are different.

[9]If punctuation can be used both with X and Y, it also interacts with type-changing rules X $\Rightarrow$ Y. Our current implementation does not deal with this case.

Functor X|Y and Y|Z combine to X|Z Our reformulation of Eisner's NF eliminates spurious ambiguities that are due to such associative composition chains. This covers not only argument clusters (which must compose), but also ambiguous cases where one constituent (e.g. Y/Z with $\alpha = \epsilon$) is the argument of the first (X/Y), and either takes the third (Z) as its own argument or is modified by the third Y\Y (there are, of course, other arrangements of such categories which are not ambiguous, e.g. X/Y Z Y\Z.

We now focus our attention on the ternary cases in which one of the constituents is a head (predicate), and the other two are either its arguments or modifiers. The counterexample to Eisner's normal-form algorithm shows that there is at least one additional kind of spurious ambiguity that arises when there are three adjacent constituents A, B, C and both A and C can compose into B. There are three cases: 1) A and C are both modifiers of B, 2) one of A or C is a modifier of B, the other is an argument of B, and 3) A and C are both arguments of B. Only 1) is a real ambiguity, but the other cases are instances of spurious ambiguity which our NF eliminates.

Argument Y, head (X\Y)/Z and argument Z combine to X In the NF derivation, the head applies first to the Z, than to Y. All other derivations are blocked, either because type-raised categories cannot apply, or because the output of composition cannot apply.

Modifier X/X, head (X|α)|β and modifier (X|α)\(X|α) combine to (X|α)|β (Fig. 4.2). This is the *"intentionally knock twice"* example. The derivations have different semantics.

Argument Y, head ((X|α)\Y)|β, and modifier X\X combine to (X|α)|β (Fig. 7). If there is an ambiguity, B must have a category of the form

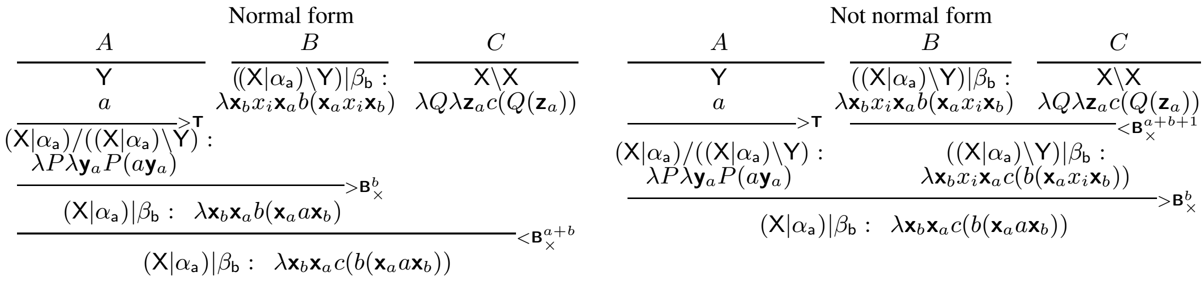


Figure 7: Argument Y, head $((\mathsf{X}|\alpha_\mathsf{a})\backslash\mathsf{Y})|\beta_\mathsf{b}$, and modifier $\mathsf{X}\backslash\mathsf{X}$ combine to $(\mathsf{X}|\alpha_\mathsf{a})|\beta_\mathsf{b}$

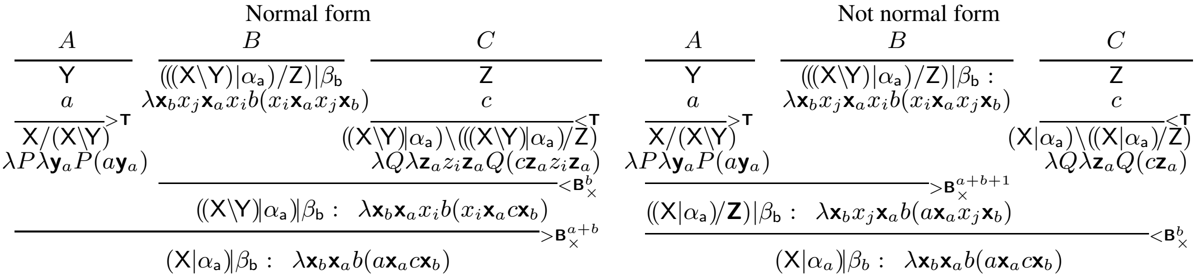


Figure 8: Argument Y, head $(((\mathsf{X}\backslash\mathsf{Y})|\alpha)/\mathsf{Z})|\beta$ and argument Z combine to $(\mathsf{X}|\alpha)|\beta$

$((\mathsf{X}|\alpha)\backslash\mathsf{Y}_\mathsf{i})|\beta$ (with X possibly complex and α, β possibly empty), and C must have a category of the form $\mathsf{X}\backslash\mathsf{X}$. We obtain the NF derivation by first combining head and argument, followed by the modifier. The other derivation violates the NF constraints.

Argument Y**, head** $(((\mathsf{X}\backslash\mathsf{Y})|\alpha)/\mathsf{Z})|\beta$ **and argument** Z **combine to** $(\mathsf{X}|\alpha)|\beta$ (Fig. 8) The derivation in which Z composes first is in NF. The derivation in which the Y combines first with the head is blocked.

Arguments Y_A**,** Y_B**, head** $(((\mathsf{X}\backslash\mathsf{Y}_1)|\alpha)\backslash\mathsf{Y}_2)|\beta$ **combine to** $(\mathsf{X}|\alpha)|\beta$ There are two readings: standard (Y_A:=Y_1, Y_B:=Y_2), and scrambled (Y_A:=Y_2, Y_B:=Y_1). If α and β are empty, function application is sufficient for the standard reading, and our NF constraint 1 excludes the 'argument cluster' derivation in which both Y_A and Y_B type-raise, compose and then apply to the head. Otherwise, at least one of the arguments has to type-raise and compose into the head. If both α and β are non-empty, each interpretation has only one derivation in which the type-raised Y_A composes into the output of the composition of the type-raised Y_B with the head. Since the degree of the second composition is lower than the first, this is allowed by our NF constraint 2.

Argument Y_A **and heads** $(((\mathsf{X}\backslash\mathsf{Y}_1)|\alpha)/\mathsf{Z}$ **and** $((\mathsf{Z}|\beta)\backslash\mathsf{Y}_2)|\gamma$ **combine to** $(((\mathsf{X}|\alpha)|\beta)\backslash\mathsf{Y}_2)|\gamma$ **or to** $(((\mathsf{X}|\backslash\mathsf{Y}_1\alpha)|\beta)|\gamma$ There are two readings: standard (Y_A:=Y_1) or scrambled (Y_A:=Y_2). Depending on the maximal degree n of $\mathbf{B}^n$ allowed by the grammar, the standard reading one can either be obtained by type-raising Y_A and composing into the first head (allowed by our NF) or by first composing the two heads and then composing the type-raised Y_A into the cluster (allowed by Eisner, but not by us). The second reading requires the heads to compose and then Y_A to apply or compose (depending on the arity of γ), which is allowed by our NF constraint 2 because the degree of this second composition is lower than that of the first.

Our NF and the bound $N_\mathbf{T}$ **on type-raising** If $\mathsf{X}\backslash\mathsf{X}$ in Fig. 7 is replaced with a (non-type-raised) category $\mathsf{Z}\backslash\mathsf{X}$ (for $\mathsf{Z} \neq \mathsf{X}$), the non-NF derivation requires $\mathbf{T}^{|Z|+a}$, whereas the NF-derivation requires $\mathbf{T}^{|X|+a}$. If we stipulate a finite bound $N_\mathbf{T}$ on the degree of type-raising, and if $|X| > |Z|$ and $|X| + a > N_\mathbf{T}$, our NF cannot be derived anymore. If such $\mathsf{Z}\backslash\mathsf{X}$ (with $\mathsf{X} \in \mathcal{C}_{arg}$) can be derived from the lexicon, our NF requires therefore a potentially unbounded degree of type-raising. The $\mathbf{T}$-degree

	Sentence length l=15...30			
	15	20	25	30
No NF (total #derivs)	4.13E6	5.66E8	3.06E11	1.59E14
Eisner **B**	18.92%	9.05%	3.63%	2.14%
Our **B**	18.38%	8.97%	3.60%	2.02%
Our **B** , **T**	2.92%	1.22%	0.37%	0.10%
Our full NF	2.60%	0.93%	0.33%	0.09%

(a) Median % of allowed derivations

	Sentence length l= 30			
	Min	Mean	Median	Max
No NF	5.99E9	8.19E15	1.59E14	2.61E17
Eisner **B**	1.60%	2.68%	2.14%	2.76%
Our **B**	1.57%	2.49%	2.02%	2.69%
Our **B** ,**T**	0.64%	0.07%	0.10%	0.05%
Our full NF	0.53%	0.06%	0.09%	0.05%

(b) Statistics on the % of allowed derivations

Figure 9: Experimental results: the effect of different normal forms on the number of derivations

of the non-NF derivation in Fig. 8 is also one less than that of the NF derivation, but its **B**-degree is increased by one, so for $N_{\mathbf{T}} = N_{\mathbf{B}}$ either both derivations are possible or neither.

What remains to be proven is that we have considered all cases of spurious ambiguity involving three constituents, and that all cases of spurious ambiguity that arise for more than three constituents reduce to these cases.

5 The effects of normal form parsing

We now illustrate the impact of the different normal form variants on a small, restricted, grammar. We define a set of atomic categories, a set of lexical categories (up to a fixed arity N_{Lex}), and compile out all possible rule instantiations (including compositions up to a fixed degree $N_{|B}$) that generate categories up to a fixed arity N_{cat}[10]

The effect of different normal forms This experiment is intended to examine how normal form parsing might reduce spurious ambiguity for actual grammars, e.g. for unsupervised estimation of stochastic CCGs. We created a small English grammar with atomic categories $\mathsf{S}, \mathsf{NP}, \mathsf{N}, \mathsf{conj}, ., , ;$ and 47 lexical categories using $N_{Lex} = 3$, $N_{\mathbf{B}} = 3$, $N_{Cat} = 15$. There are two type-changing rules ($\mathsf{N} \Rightarrow \mathsf{NP}$ and $\mathsf{S/NP} \Rightarrow \mathsf{NP\backslash NP}$). We accept derivations of S, NP and $\mathsf{S\backslash NP}$. The $\mathsf{T|X}$ in **T** has to be a lexical category. Our lexical categories are divided into disjoint sets of adjuncts of the form $\mathsf{X|X}$ and $\mathsf{(X|X)|Y}$, head (both atomic and complex), and punctuation and conjunction categories. The comma can act as a conjunction or to set off modifiers (requiring punctuation rules of the form $\mathsf{X|X}\ , \Rightarrow \mathsf{X|X}$ and $,\ \mathsf{X|X} \Rightarrow \mathsf{X|X}$). We furthermore define coarse-grained parts of speech (noun, verb, function word, conj, other) and decide for each part of speech which lexical categories it can take. We compare different NF settings for sentences of lengths 15–30 from Europarl (Koehn, 2005). At each length, we compare 100 sentences that our grammar can parse. All NFs can parse all sentences the full grammar can parse. Results (Fig. 9(a)) show that our NF reduces the number of derivations significantly over Eisner's NF, even though our (full) grammar only allows a restricted set of type-raising rules. Fig. 9(b) illustrates the combinatorial explosion of spurious derivations as the sentence length increases.

[10]The restriction of categories to a fixed arity means that we could generate cross-serial dependencies $N_1...N_nV_1...V_n$ only up to $n = A_{cat}$.

6 Conclusions

We have proposed a modification and extension of Eisner (1996)'s normal form that is more appropriate for commonly used variants of CCG with grammatical type-raising and generalized composition of bounded degree, as well as some non-combinatory extensions to CCG. Our experiments indicate that incorporating normal form constraints to deal with grammatical type-raising drastically reduces the number of derivations. We have sketched the outline of a proof that our normal form is safe and complete for the variant of CCG we consider, althoug we have seen that under certain circumstances, type-raising of unbounded degree may be required. Future work will investigate this issue further, and will also aim to turn our informal arguments about the adequacy of our approach into a full proof, and provide more experiments on a wider range of grammars and languages.

References

Boxwell, Stephen, Dennis Mehay, and Chris Brew. 2009. Brutus: A semantic role labeling system incorporating CCG, CFG, and dependency features. In *Proceedings of the 47th ACL/4th IJCNLP*, pages 37–45.

Clark, Stephen and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Curran, James, Stephen Clark, and Johan Bos. 2007. Linguistically motivated large-scale NLP with C&C and boxer. In *Proceedings of the 45th ACL Companion Volume (Demo and Poster Sessions)*, pages 33–36, Prague, Czech Republic.

Eisner, Jason. 1996. Efficient normal-form parsing for Combinatory Categorial Grammar. In *Proceedings of the 34th ACL*, pages 79–86, Santa Cruz, CA.

Espinosa, Dominic, Michael White, and Dennis Mehay. 2008. Hypertagging: Supertagging for surface realization with CCG. In *Proceedings of ACL-08: HLT*, pages 183–191, Columbus, Ohio.

Gildea, Daniel and Julia Hockenmaier. 2003. Identifying semantic roles using Combinatory Categorial Grammar. In *Proceedings of EMNLP*, Sapporo, Japan.

Hepple, Mark and Glyn Morrill. 1989. Parsing and derivational equivalence. In *Proceedings of the Fourth EACL*, pages 10–18, Manchester, UK.

Hockenmaier, Julia and Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the penn treebank. *Computational Linguistics*, 33(3):355–396.

Hockenmaier, Julia. 2003. *Data and models for statistical parsing with Combinatory Categorial Grammar*. Ph.D. thesis, School of Informatics, University of Edinburgh.

Hoffman, Beryl. 1995. *Computational Analysis of the Syntax and Interpretation of 'Free' Word-order in Turkish*. Ph.D. thesis, University of Pennsylvania. IRCS Report 95-17.

Hoyt, Frederick and Jason Baldridge. 2008. A logical basis for the D combinator and normal form in CCG. In *Proceedings of ACL-08: HLT*, pages 326–334, Columbus, Ohio.

Karttunen, Lauri. 1989. Radical lexicalism. In Baltin, M.R. and A.S. Kroch, editors, *Alternative Conceptions of Phrase Structure*. Chicago University Press, Chicago.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *10th MT Summit*, pages 79–86, Phuket, Thailand.

Komagata, Nobo. 1997. Generative power of CCGs with generalized type-raised categories. In *ACL35/EACL8 (Student Session)*, pages 513–515.

Komagata, Nobo. 2004. A solution to the spurious ambiguity problem for practical combinatory categorial grammar parsers. *Computer Speech & Language*, 18(1):91 – 103.

Niv, Michael. 1994. A psycholinguistically motivated parser for CCG. In *Proceedings of The 32nd ACL*, Las Cruces, NM, pages 125–132.

Pareschi, Remo and Mark Steedman. 1987. A lazy way to chart parse with categorial grammars. In *Proceedings of the 25th ACL*, pages 81–88, Stanford, CA.

Shieber, Stuart M., Yves Schabes, and Fernando C. N. Pereira. 1995. Principles and implementation of deductive parsing. *Journal of Logic Programming*, 24(1–2):3–36, July–August.

Steedman, Mark. 2000. *The Syntactic Process*. MIT Press, Cambridge, MA.

Vijay-Shanker, K and David J Weir. 1993. Parsing some constrained grammar formalisms. *Computational Linguistics*, 19(4):591–636.

Weir, David. 1988. *Characterising Mildly Context-sensitive Grammar Formalisms*. Ph.D. thesis, University of Pennsylvania. Tech. Report CIS-88-74.

Wittenburg, Kent B. 1986. *Natural Language Parsing with Combinatory Categorial Grammar in a Graph-Unification Based Formalism*. Ph.D. thesis, University of Texas at Austin.

Zettlemoyer, Luke S. and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of the 21st UAI*, pages 658–666, Edinburgh, UK.

Zettlemoyer, Luke and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of EMNLP-CoNLL*, pages 678–687, Prague, Czech Republic.

Acknowledgements

We would like to thank Mark Steedman for helpful discussions, and Jason Eisner for his very generous feedback which helped to greatly improve this paper. All remaining errors and omissions are our own responsibility. J.H is supported by NSF grant IIS 08- 03603 INT2-Medium.

An Empirical Study on Web Mining of Parallel Data

Gumwon Hong[1], Chi-Ho Li[2], Ming Zhou[2] and Hae-Chang Rim[1]

[1]Department of Computer Science & Engineering, Korea University
{gwhong,rim}@nlp.korea.ac.kr

[2]Natural Language Computing Group, Microsoft Research Asia
{chl,mingzhou}@microsoft.com

Abstract

This paper[1] presents an empirical approach to mining parallel corpora. Conventional approaches use a readily available collection of comparable, non-parallel corpora to extract parallel sentences. This paper attempts the much more challenging task of directly searching for high-quality sentence pairs from the Web. We tackle the problem by formulating good search query using 'Learning to Rank' and by filtering noisy document pairs using IBM Model 1 alignment. End-to-end evaluation shows that the proposed approach significantly improves the performance of statistical machine translation.

1 Introduction

Bilingual corpora are very valuable resources in NLP. They can be used in statistical machine translation (SMT), cross language information retrieval, and paraphrasing. Thus the acquisition of bilingual corpora has received much attention.

Hansards, or parliamentary proceedings in more than one language, are obvious source of bilingual corpora, yet they are about a particular domain and therefore of limited use. Many researchers then explore the Web. Some approach attempts to locate bilingual text within a web page (Jiang et al., 2009); some others attempt to collect web pages in different languages and decide the parallel relationship between the web pages by means of *structural cues*, like existence of a common ancestor web page, similarity between URLs, and similarity between the HTML structures (Chen and Nie, 2000; Resnik and Smith, 2003; Yang and Li, 2003; Shi et al., 2006). The corpora thus obtained are generally of high quality and wide variety in domain, but the amount is still limited, as web pages that exhibit those structural cues are not abundant.

Some other effort is to mine bilingual corpora by *textual means* only. That is, two pieces of text are decided to be parallel merely from the linguistic perspective, without considering any hint from HTML markup or website structure. These approaches (Zhao and Vogel, 2002; Utiyama and Isahara 2003; Fung and Cheung, 2004; Munteanu and Marcu, 2005; Abdul-Rauf and Schwenk, 2009) share roughly the same framework:

Phase 1: Document Pair Retrieval

1) documents in some target language (TL) are stored in some database;
2) each document in some source language (SL) is represented by some TL keywords;
3) the TL keywords in (2) are used to assign some TL documents to a particular SL document, using some information retrieval (IR) technique. For example, Munteanu and Marcu (2005) apply the Lemur IR toolkit, Utiyama and Isahara (2003) use the BM25 similarity measure, and Fung and Cheung (2004) use cosine similarity. Each TL document pairs up with the SL document to form a candidate parallel document pair.

Phase 2: Sentence Pair Extraction

1) sentence pairs can be obtained by running sentence alignment over all candidate document pairs (or a selection of them) (Zhao and Vogel, 2002; Utiyama and Isahara, 2003);
2) sentence pairs can also be selected, by some classifier or reliability measure, from the candidate sentence pairs enumerated from the candidate document pairs (Munteanu and Marcu, 2005).

Note that the primary interest of these approaches is sentence pairs rather than document

[1] This work has been done while the first author was visiting Microsoft Research Asia.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 474–482,
Beijing, August 2010

pairs, partially because document pair retrieval is not accurate, and partially because the ultimate purpose of these corpora is SMT training, which is based on sentence pairs. It is found that most of the sentence pairs thus obtained are not truly parallel; rather they are loose translations of each other or they carry partially similar messages. Such bilingual corpora are thus known as *comparable corpora*, while genuinely mutual translations constitute *parallel corpora*.

Note also that all these comparable corpus mining approaches are tested on *closed document collections* only. For example, Zhao and Vogel (2002), Utiyama and Isahara (2003), and Munteanu and Marcu (2005) all acquire their comparable corpora from a collection of news articles which are either downloaded from the Web or archived by LDC. The search of candidate document pairs in such a closed collection is easy in three ways:

1) all the TL documents come from the same news agency and they are not mixed up with similar documents from other news agencies;
2) all the TL documents are news text and they are not mixed up with text of other domains;
3) in fact, the search in these approaches is made easier by applying tricks like date window.

There is no evidence that these methods apply to corpus mining from an open document collection (e.g. the entire Web) without search constraint. The possibility of open-ended text mining is a crucial problem.

This paper focuses on bilingual corpus mining using only textual means. It attempts to answer two questions:

1) Can comparable corpus mining be applied to an open document collection, i.e., the Web?
2) Can comparable corpus mining be adapted to parallel corpus mining?

We give affirmation to both questions. For the first problem, we modify document pair retrieval so that there is no longer a closed set of TL documents. Instead we search for candidate TL documents for a particular SL document from the Web by means of some Web search engine. For the second problem, in Phase 2 we replace the sentence pair classifier by a document pair filter and a sentence alignment module. Based on end-to-end SMT experiments, we will show that 1) high quality bilingual corpora can be mined from the Web; 2) the very first key to Web-mining of bilingual corpus is the formulation of good TL keywords to represent a SL document; 3) a simple document pair filter using IBM Model 1 probabilities is able to identify parallel corpus out of noisy comparable text; and 4) Web-mined parallel corpus, despite its smaller size, improves SMT much more than Web-mined comparable corpus.

2 Problem Setting

Our ultimate goal is to mine from the Web training data for translation from Chinese (SL) to English (TL). As the first step, about 11,000 Chinese web pages of news articles are crawled from some Chinese News sites. Then the task is to search for the English sentences corresponding to those in the selected SL articles. These selected SL news articles all contain cue phrases like “根据外电报道” (*according to foreign media*), as these cue phrases suggest that the Chinese articles are likely to have English counterparts. Moreover, each selected SL article has at least 500 words (empirically determined) since we assume that it is much easier to formulate reliable keywords from a long document than a short one.

3 Document Pair Retrieval

Conventional approaches to comparable corpus mining usually start with document pair retrieval, which assigns to each SL document a set of candidate TL documents. This step is essentially a preliminary search for candidate sentence pairs for further scrutiny in Phase 2. The target is to find document pairs which may contain many good sentence pairs, rather than to discard document pairs which may not contain good sentence pairs. Therefore, *recall is much more emphasized* than precision.

Document pair retrieval in conventional approaches presumes a closed set of TL documents which some IR system can handle easily. In this paper we override this presumption and attempt a much more challenging retrieval task, viz. to search for TL documents among the Web, using the search engines of Google and Yahoo. Therefore we are subject to a much noisier data domain. The correct TL documents may not be indexed by the search engines at all, and even when the target documents are indexed, it requires a more sophisticated formulation of queries to retrieve them.

In response to these challenges, we propose various kinds of queries (elaborated in the following subsections). Moreover, we merge the TL documents found by each query into a big collection, so as to boost up the recall. In case a query fails to retrieve any document, we iteratively drop a keyword in the query until some documents are found. On the other hand, although the document pairs in question are of news domain, we use the general Google/Yahoo web search engines instead of the specific news search engines, because 1) the news search engines keep only a few web pages for all pages about the same news event, and 2) we leave open possibility for correct TL documents to be found in non-news web pages.

3.1 Simple Queries

There are three baseline formulations of queries:

1) Query of translations of SL TF-IDF-ranked keywords ($Q_{SL\text{-}TFIDF}$). This is the method proposed by Munteanu and Marcu (2005). All the words in a SL document are ranked by TF-IDF and the top-N words are selected. Each keyword is then translated into a few TL words by a statistically learned dictionary. In our experiments the dictionary is learned from NIST SMT training data.
2) Query of TF-IDF-ranked machine translated keywords ($Q_{TL\text{-}TFIDF}$). It is assumed that a machine translation (MT) system is better at handling lexical ambiguity than simple dictionary translation. Thus we propose to first translate the SL document into TL and extract the top-N TF-IDF-ranked words as query. In our experiments the MT system used is hierarchical phrase-based system (Chiang, 2007).[2]
3) Query of named entities (Q_{NE}). Another way to tackle the drawback of $Q_{SL\text{-}TFIDF}$ is to focus on named entities (NEs) only, since NEs often provide strong clue for identifying correspondence between two languages. All NEs in a SL document are ranked by TF-IDF, and the top-N NEs are then translated (word by word) by dictionary. In our experiments we identify SL (Chinese) NEs implicitly found by the word segmentation algorithm stated in Gao et al. (2003), and the dictionaries for translating NEs include the same one used for $Q_{SL\text{-}TFIDF}$, and the LDC Chinese/English NE dictionary. For the NEs not covered by our dictionary, we use Google translation service as a back-up.

A small-scale experiment is run to evaluate the merits of these queries. 300 Chinese news web pages in three different periods (each 100) are collected. For each Chinese text, each query (containing 10 keywords) is constructed and submitted to both Google and Yahoo Search, and top-40 returned English web pages for each search are kept. Note that the Chinese news articles are not part of 11,000 pages in section 2. In fact, they do not only satisfy the requirement of length and cue phrases (described in section 2), but they also have another property that they are translated from some English news articles (henceforth target pages) on the Web. Thus they are ideal data for studying the performance of document pair retrieval.

To test the influence of translation quality in document pair retrieval, we also try 'oracle queries', i.e. queries formulated directly from the target pages:

1) OQ_{TFIDF}. This is the query of the top-N TF-IDF-ranked words from the target page.
2) OQ_{NE}. This is the query of the top-N TF-IDF-ranked NEs from the target web page.

We define recall as the proportion of SL documents whose true target pages are found. The comparison between a retrieved page and the target page is done by Longest Common Subsequence (LCS) ratio, defined as the length of the longest common word sequence of two documents divided by the length of the longer of two documents. The threshold 0.7 is adopted as it is strict enough to distinguish parallel document pairs from non-parallel ones.

Table 1 shows the recalls for various queries. It can be seen from Tests 6 and 7 that the largest recall, 85% (within top 40 search results), is achieved when the word distributions in the target web pages are known. In the real scenario where the true English word distribution is not known, the recalls achieved by the simple queries are very unsatisfactory, as shown by Tests 1 to 3. This clearly shows how challenging Web-based mining of bilingual corpora is. Another challenge can be observed in comparing across

[2] We also try online Google translation service, and the performance was roughly the same.

ID	Query	Remote	Near	Recent
1	$Q_{SL\text{-}TFIDF}$	7	6	8
2	$Q_{TL\text{-}TFIDF}$	16	19	32
3	Q_{NE}	16	21	38
4	union(2,3)	27	31	48
5	union(1,2,3)	28	31	48
6	OQ_{TFIDF}	56	66	82
7	OQ_{NE}	62	68	85
8	$Overlap_{TFIDF}$	52	51	74
9	$Overlap_{NE}$	55	62	83

Table 1: Recall (%age) of simple queries. 'Remote' refers to news documents more than a year ago; 'Near' refers to documents about 3 months ago; 'Recent' refers to documents in the last two weeks.

columns, viz. it is much more difficult to retrieve outdated news document pairs. This implies that bilingual news mining must be incrementally carried out.

Comparing Test 1 to Tests 2 and 3, it is obvious that $Q_{SL\text{-}TFIDF}$ is not very useful in document pair retrieval. This confirms our hypothesis that suitable TL keywords are not likely to be obtained by simple dictionary lookup. While the recalls by $Q_{TL\text{-}TFIDF}$ are similar to those by Q_{NE}, the two queries contribute in different ways. Test 4 simply merges the Web search results in Tests 2 and 3. The significantly higher recalls in Test 4 imply that each of the two queries finds substantially different targets than each other. The comparison of Test 5 to Test 4 further confirms the weakness of $Q_{SL\text{-}TFIDF}$.

The huge gap between the three simple queries and the oracle queries shows that the quality of translation of keywords from SL to TL is a major obstacle. There are two problems in translation quality: 1) the MT system or dictionary *cannot produce any translation* for a SL word (let us refer to such TL keywords as 'Utopian translations'); 2) the MT system or dictionary *produces an incorrect translation* for a SL word. We can do very little for the Utopian translations, as the only solution is simply to use a better MT system or a larger dictionary. On the contrary, it seems that the second problem can somewhat be alleviated, if we have a way to distinguish those terms that are likely to be correct translations from those terms that are not. In other words, it may be worthwhile to reorder candidate TL keywords by our confidence in its translation quality.

Tests 8 and 9 in Table 1 show that this hypothesis is promising. In both tests the TF-IDF-based (Test 8) or the NE-based (Test 9) keywords are selected from only those TL words that appear both in the target page and the machine translated text of the source page. In other words, we ensure that the keywords in the query must be correct translations. The recalls (especially the recalls by NE-based query in Test 9) are very close to the recalls by oracle queries. The conclusion is, even though we cannot produce the Utopian translations, document pair retrieval can be improved to a large extent by removing incorrect translations. Even an imperfect MT system or NE dictionary can help us achieve as good document pair retrieval recall as oracle queries.

In the next subsection we will take this insight into our bilingual data mining system, by selecting keywords which are likely to be correct translation.

3.2 Re-ranked Queries

Machine learning is applied to re-rank keywords for a particular document. The re-ranking of keywords is based on two principles. The first one is, of course, the confidence on the translation quality. The more likely a keyword is a correct translation, the higher this keyword should be ranked. The second principle is the representativeness of document. The more representative of the topic of the document where a keyword comes from, the higher this keyword should be ranked. The design of features should incorporate both principles.

The representativeness of document is manifested in the following features for each keyword per each document:

- *TF*: the term frequency.
- *IDF*: the inverted document frequency.
- *TF-IDF*: the product of *TF* and IDF.
- *Title word*: it indicates whether a keyword appears in the title of the document.
- *Bracketed word*: it indicates whether a word is enclosed in a bracket in the source document.
- *Position of first appearance*: the position where a keyword first appears in a document, normalized by number of words in the document.

- *NE types*: it indicates whether a keyword is a person, organization, location, numerical expression, or non NE.

The confidence on translation quality is manifested in the following features:

- *Translation source*: it indicates whether the keyword (in TL) is produced by MT system, dictionary, or by both.
- *Original word*: it indicates whether the keyword is originally written in *English* in the source document. Note that this feature also manifests the representativeness of a document.
- *Dictionary rank*: if the keyword is a NE produced by dictionary, this feature indicates the rank of the NE keyword among all translation options registered in the dictionary.

It is difficult to definitely classify a TL keyword into good or bad translation in absolute sense, and therefore we take the alternative of ranking TL keywords with respect to the two principles. The learning algorithm used is Ranking SVM (Herbrich et al., 2000; Joachims, 2006), which is a state-of-the-art method of the "Learning to rank" framework.

The training dataset of the keyword re-ranker comprises 1,900 Chinese/English news document pairs crawled from the Web[3]. This set is not part of 11,000 pages in section 2. These document pairs share the same properties as those 300 pairs used in Section 3.1. For each English/target document, we build a set T_{ALL}, which contains all words in the English document, and also a set T_{NE}, which is a subset of T_{ALL} such that all words in T_{NE} are NEs in T_{ALL}. The words in both sets are ranked by TFIDF. On the other hand, for each Chinese/source document, we machine-translate it and then store the translated words into a set S, and we also add the dictionary translations of the source NEs into S. Note that S is composed of both good translations (appearing in the target document) and bad translations (not appearing in the target document).

Then there are two ways to assign labels to the words in S. In the first way of labeling (L_{ALL}), the label ***3*** is assigned to those words in S which are ranked among top 5 in T_{ALL}, label ***2*** to those ranked among top 10 but not top 5 in T_{ALL}, ***1*** to those beyond top 10 but still in T_{ALL}, and ***0*** to those words which do not appear in T_{ALL} at all. The second way of labeling, L_{NE}, is done in similar way with respect to T_{NE}. Collecting all training samples over all document pairs, we can train a model, M_{ALL}, based on labeling L_{ALL}, and another model M_{NE}, based on labeling L_{NE}.

The trained models can then be applied to re-rank the keywords of simple queries. In this case, a set S_{TEST} is constructed from the 300 Chinese documents in similar way of constructing S. We repeat the experiment in Section 3.1 with two new queries:

1) $Q_{RANK\text{-}TFIDF}$: the top N keywords from re-ranking S_{TEST} by M_{ALL};
2) $Q_{RANK\text{-}NE}$: the top N keywords from reranking S_{TEST} by M_{NE}.

Again N is chosen as 10.

ID	Query	Remote	Near	Recent
10	$Q_{RANK\text{-}TFIDF}$	18	20	29
11	$Q_{RANK\text{-}NE}$	35	43	54
12	union(10,11)	39	49	63

Table 2: Recall (%age) of re-ranked queries.

The results shown in Table 2 indicate that, while the re-ranked queries still perform much poorer than oracle queries (Tests 6 and 7 in Table 1), they show great improvement over the simple queries (Tests 1 to 5 in Table 1). The results also show that re-ranked queries based on NEs are more reliable than those based on common words.

4 Sentence pair Extraction

The document pairs obtained by the various queries described in Section 3 are used to produce sentence pairs as SMT training data. There are two different methods of extraction for corpora of different nature.

4.1 For Comparable Corpora

Sentence pair extraction for comparable corpus is the same as that elaborated in Munteanu and Marcu (2005). All possible sentence pairs are enumerated from all candidate document pairs produced in Phase 1. These huge number of candidate sentence pairs are first passed to a coarse sentence pair filter, which discards very unlikely candidates by heuristics like sentence

[3] We also attempt to add more training data for re-ranking but the performance remain the same.

length ratio and percentage of word pairs registered in some dictionary.

The remaining candidates are then given to a Maximum Entropy based classifier (Zhang, 2004), which uses features based on alignment patterns produced by some word alignment model. In our experiment we use the HMM alignment model with the NIST SMT training dataset. The sentence pairs which are assigned as positive by the classifier are collected as the mined comparable corpus.

4.2 For Parallel Corpora

The sentence pairs obtained in Section 4.1 are found to be mostly not genuine mutual translations. Often one of the sentences contains some extra phrase or clause, or even conveys different meaning than the other. It is doubtful if the document pairs from Phase 1 are too noisy to be processed by the sentence pair classifier. An alternative way for sentence pair extraction is to further filter the document pairs and discard any pairs that do not look like parallel.

It is hypothesized that the parallel relationship between two documents can be assimilated by the word alignment between them. The document pair filter produces the Viterbi alignment, with the associated probability, of each document pair based on IBM Model 1 (Brown et al., 1993). The word alignment model (i.e. the statistical dictionary used by IBM Model 1) is trained on the NIST SMT training dataset. The probability of the Viterbi alignment of a document pair is the sole basis on which we decide whether the pair is genuinely parallel. That is, an empirically determined threshold is used to distinguish parallel pairs from non-parallel ones. In our experiment, a very strict threshold is selected so as to boost up the precision at the expense of recall.

There are a few important details that enable the document pair filter succeed in identifying parallel text:

1) Function words and other common words occur frequently and so any pair of common word occupies certain probability mass in an alignment model. These common words enable even non-parallel documents achieve high alignment probability. In fact, it is well known that the correct alignment of common words must take into account positional and/or structural factors, and it is beneficial to a simple alignment model like IBM Model 1 to work on data without common words. Therefore, all words on a comprehensive stopword list must be removed from a document pair before word alignment.
2) The alignment probability must be normalized with respect to sentence length, so that the threshold applies to all documents regardless of document length.

Subjective evaluation on selected samples shows that most of the document pairs kept by the filter are genuinely parallel. Thus the document pairs can be broken down into sentence pairs simply by a sentence alignment method. For the sentence alignment, our experiments use the algorithm in Moore (2002).

5 Experiments

It is a difficult task to evaluate the quality of automatically acquired bilingual corpora. As our ultimate purpose of mining bilingual corpora is to provide more and better training data for SMT, we evaluate the parallel and comparable corpora with respect to improvement in Bleu score (Papineni et al., 2002).

5.1 Experiment Setup

Our experiment starts with the 11,000 Chinese documents as described in Section 2. We use various combinations of queries in document pair retrieval (Section 3). Based on the candidate document pairs, we produce both comparable corpora and parallel corpora using sentence pair extraction (Section 4). The corpora are then given to our SMT systems as training data.

The SMT systems are our implementations of phrase-based SMT (Koehn et al., 2003) and hierarchical phrase-based SMT (Chiang, 2007). The two systems employ a 5-gram language model trained from the Xinhua section of the Gigaword corpus. There are many variations of the bilingual training dataset. The B1 section of the NIST SMT training set is selected as the baseline bilingual dataset; its size is of the same order of magnitude as most of the mined corpora so that the comparison is fair. Each of the mined bilingual corpora is compared to that baseline dataset, and we also evaluate the performance of the combination of each mined bilingual corpus with the baseline set.

Bilingual Training Corpus	Phrase-based SMT (PSMT)		Hierarchical PSMT	
	NIST 2005	NIST 2008	NIST 2005	NIST 2008
B1 (baseline)	33.08	21.66	32.85	21.18
B1+comparable(M&M)	33.51(+0.43)	22.71(+1.05)	32.99(+0.14)	22.11(+0.93)
B1+comparable($Q_{RANK-NE}$)	**34.81(+1.73)**	23.30(+1.64)	34.43(+1.58)	22.85(+1.67)
B1+comparable(all simple)	34.74(+1.66)	**23.48(+1.82)**	34.28(+1.43)	**23.18(+2.00)**
B1+comparable(all ranked)	34.79(+1.71)	**23.48(+1.82)**	34.37(+1.52)	23.06(+1.88)
B1+comparable(all query)	34.74(+1.66)	23.19(+1.53)	**34.46(+1.61)**	23.12(+1.94)
B1+parallel($Q_{RANK-NE}$)	34.75(+1.67)	23.37(+1.71)	34.24(+1.39)	23.45(+2.27)
B1+parallel(all simple)	34.99(+1.91)	23.96(+2.30)	34.94(+2.09)	23.35(+2.17)
B1+parallel(all ranked)	34.76(+1.68)	23.41(+1.75)	34.54(+1.69)	23.59(+2.41)
B1+parallel(all query)	**35.40(+2.32)**	**23.47(+1.81)**	**35.27(+2.42)**	**23.61(+2.43)**

Table 4: Evaluation of translation quality improvement by mined corpora. The figures inside brackets refer to the improvement over baseline. The bold figures indicate the highest Bleu score in each column for comparable corpora and parallel corpora, respectively.

The SMT systems learn translation knowledge (phrase table and rule table) in standard way. The parameters in the underlying log-linear model are trained by Minimum Error Rate Training (Och, 2003) on the development set of NIST 2003 test set. The quality of translation output is evaluated by case-insensitive BLEU4 on NIST 2005 and NIST 2008 test sets[4].

5.2 Experimental result

Table 3 lists the size of various mined parallel and comparable corpora against the baseline B1 bilingual dataset. It is obvious that for a specific type of query in document pair retrieval, the parallel corpus is significantly smaller than the corresponding comparable corpus.

The apparent explanation is that a lot of document pairs are discarded due to the document pair filter. Note that the big difference in size of the two comparable corpora by single queries, i.e., $Q_{RANK-NE}$ and M&M, verifies again that re-ranked queries based on NEs are more reliable in sentence pair extraction.

Queries	SP extraction	#SP	#SL words	#TL words
Baseline: B1 in NIST		68K	1.7M	1.9M
M&M	comparable	43K	1.1M	1.2M
$Q_{RANK-NE}$	comparable	98K	2.7M	2.8M
all simple	comparable	98K	2.6M	2.9M
all ranked	comparable	115K	3.1M	3.3M
all query	comparable	135K	3.6M	4.0M
$Q_{RANK-NE}$	parallel	66K	1.9M	1.8M
all simple	parallel	52K	1.5M	1.4M
all ranked	parallel	73K	2.1M	2.0M
all query	parallel	90K	2.5M	2.4M

Table 3: Statistics on corpus size. SP means sentence pair. 'all simple', 'all ranked', and 'all query' refer to the merge of the retrieval results of all simple queries, all re-ranked queries, and all simple and re-ranked queries, respectively; M&M (after Munteanu and Marcu (2005)) refers to $Q_{SL-TFIDF}$.

Table 4 lists the Bleu scores obtained by *augmenting* the baseline bilingual training set *with* the mined corpora. The most important observation is that, despite their smaller size, parallel corpora lead to no less, and often better, improvement in translation quality than comparable corpora. That is especially true for the case where document pair retrieval is based on all five types of query[5]. The superiority of parallel corpora confirms that, in Phase 2 (sentence pair extraction), quality is more important than quantity and thus the filtering of document pair/sentence pair must not be generous.

On the other hand, sentence pair extraction for parallel corpora generally achieves the best result when all queries are applied in document pair retrieval. It is not sufficient to use the more sophisticated re-ranked queries. That means in Phase 1 quantity is more important and we must seek more ways to retrieve as many document pairs as possible. That also confirms the emphasis on recall in document pair retrieval.

Looking into the performance of comparable corpora, it is observed that the M&M query does not effectively apply to Web mining of comparable corpora but the proposed queries do. Any of the proposed query leads to better result than the conventional method, i.e. M&M. Moreover, it can be seen that all four combinations of proposed queries achieve similar per-

[4] It is checked that there is no sentence in the test sets overlapping with any sentences in the mined corpus.

[5] $Q_{SL-TFIDF}$, $Q_{TL-TFIDF}$, Q_{NE}, $Q_{RANK-TFIDF}$, and $Q_{RANK-NE}$

Bilingual Training Corpus	Phrase-based SMT		Hierarchical PSMT	
	NIST 2005	NIST 2008	NIST 2005	NIST 2008
B1 (baseline)	33.08	21.66	32.85	21.18
comparable(M&M)	20.84(-12.24)	14.33(-7.33)	20.65(-12.20)	13.73(-7.45)
comparable($Q_{RANK\text{-}NE}$)	26.78(-6.30)	18.54(-3.12)	27.10(-5.75)	18.02(-3.16)
comparable(all simple)	26.39(-6.69)	18.52(-3.14)	26.40(-6.45)	18.22(-2.96)
comparable(all ranked)	27.36(-5.72)	18.89(-2.77)	27.40(-5.45)	18.72(-2.46)
comparable(all query)	**27.96(-5.12)**	**19.27(-2.39)**	**27.83(-5.02)**	**19.46(-1.72)**
parallel($Q_{RANK\text{-}NE}$)	26.37(-6.71)	18.70(-2.96)	26.47(-6.38)	18.51(-2.67)
parallel(all simple)	25.65(-7.43)	18.69(-2.97)	25.28(-7.57)	18.55(-2.63)
parallel(all ranked)	26.86(-6.22)	18.94(-2.72)	27.10(-5.75)	18.78(-2.40)
parallel(all query)	**27.58(-5.50)**	**19.73(-1.93)**	**28.10(-4.75)**	**19.52(-1.66)**

Table 5: Evaluation of translation quality by mined corpora.

formance. This illustrates a particular advantage of using a single re-ranked query, viz. $Q_{RANK\text{-}NE}$, because it significantly reduces the retrieval time and downloading space required for document pair retrieval as it is the main bottleneck of whole process.

Table 5 lists the Bleu scores obtained by *replacing* the baseline bilingual training set *with* the mined corpora. It is easy to note that translation quality drops radically by using mined bilingual corpus alone. That is a natural consequence of the noisy nature of Web mined data. We should not be too pessimistic about Web mined data, however. Comparing the Bleu scores for NIST 2005 test set to those for NIST 2008 test set, it can be seen that the reduction of translation quality for the NIST 2008 set is much smaller than that for the NIST 2005 set. It is not difficult to explain the difference. Both the baseline B1 training set and the NIST 2005 comprise news wire (in-domain) text only. Although the acquisition of bilingual data also targets news text, the noisy mined corpus can never compete with the well prepared B1 dataset. On the contrary, the NIST 2008 test set contains a large portion of out-of-domain text, and so the B1 set does not gain any advantage over Web mined corpora. It might be that better and/or larger Web mined corpus achieves the same performance as manually prepared corpus.

Note also that the reduction in Bleu score by each mined corpus is roughly the same as that by each other, while in general parallel corpora are slightly better than comparable corpora.

6 Conclusion and Future Work

In this paper, we tackle the problem of mining parallel sentences directly from the Web as training data for SMT. The proposed method essentially follows the corpus mining framework by pioneer work like Munteanu and Marcu (2005). However, unlike those conventional approaches, which work on closed document collection only, we propose different ways of formulating queries for discovering parallel documents over Web search engines. Using learning to rank algorithm, we re-rank keywords based on representativeness and translation quality. This new type of query significantly outperforms existing query formulation in retrieving document pairs. We also devise a document pair filter based on IBM model 1 for handling the noisy result from document pair retrieval. Experimental results show that the proposed approach achieves substantial improvement in SMT performance.

For mining news text, in future we plan to apply the proposed approach to other language pairs. Also, we will attempt to use meta-information implied in SL document, such as “publishing date” or “news agency name”, as further clue to the document pair retrieval. Such meta-information may likely to increase the precision of retrieval, which is important to the efficiency of the retrieval process.

An important contribution of this work is to show the possibility of mining text other than news domain from the Web, which is another piece of future work. The difficulty of this task should not be undermined, however. Our success in mining news text from the Web depends on the cue phrases available in news articles. These cue phrases more or less indicate the existence of corresponding articles in another language. Therefore, to mine non-news corpus, we should carefully identify and select cue phrases.

References

Abdul-Rauf, Sadaf and Holger Schwenk. 2009. Exploiting Comparable Corpora with TER and TERp. In *Proceedings of ACL-IJCNLP 2009 workshop on Building and Using Comparable Corpora*, pages 46–54.

Brown, Peter F., Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2): 263-311.

Chen, Jiang and Jian-Yun Nie. 2000. Automatic Construction of Parallel Chinese-English Corpus for Cross-Language Information Retrieval. In *Proceedings of NAACL-ANLP*, pages 21-28.

Chiang, David. 2007. Hierarchical Phrase-based Translation. *Computational Linguistics*, 33(2): 202-228.

Fung, Pascale, and Percy Cheung. 2004. Mining very non-parallel corpora: Parallel sentence and lexicon extraction via bootstrapping and EM. In *Proceedings of 2004 Conference on Empirical Methods in Natural Language Processing*, pages 57-63.

Gao, Jianfeng, Mu Li, and Changning Huang. 2003. Improved Source-Channel Models for Chinese Word Segmentation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 272-279.

Herbrich, Ralf, Thore Graepel, and Klaus Obermayer. 2000. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, pages 115–132. MIT Press, Cambridge, MA.

Jiang, Long, Shiquan Yang, Ming Zhou, Xiaohua Liu, and Qingsheng Zhu. 2009. Mining Bilingual Data from the Web with Adaptively Learnt Patterns. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and 4th International Joint Conference on Natural Language Processing*, pages 870-878.

Joachims, Thorsten. 2006. Training Linear SVMs in Linear Time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 217-226.

Koehn, Philipp, Franz Och, and Daniel Marcu. 2003. Statistical Phrase-based Translation. In *Proceedings of conference combining Human Language Technology conference series and the North American Chapter of the Association for Computational Linguistics conference series*, pages 48-54.

Moore, Robert. 2002. Fast and Accurate Sentence Alignment of Bilingual Corpora. In *Proceedings of the 5th conference of the Association for Machine Translation in the Americas*, pages 135–144.

Munteanu, Dragos, and Daniel Marcu. 2005. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *Computational Linguistics*, 31(4): 477-504.

Och, Franz J. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160-167.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311-318.

Resnik, Philip, and Noah Smith. 2003. The Web as a Parallel Corpus. *Computational Linguistics*, 29(3): 349-380.

Shi, Lei, Cheng Niu, Ming Zhou, and Jianfeng Gao. 2006. A DOM Tree Alignment Model for Mining Parallel Data from the Web. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 489-496.

Utiyama, Masao, and Hitoshi Isahara. 2003. Reliable Measures for Aligning Japanese-English News Articles and Sentences. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 72-79.

Vogel, Stephan. 2003. Using noisy bilingual data for statistical machine translation. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 175-178.

Yang, Christopher C., and Kar Wing Li. 2003. Automatic construction of English/Chinese parallel corpora. *Journal of the American Society for Information Science and Technology*, 54(8):730–742.

Zhang, Le. 2004. Maximum Entropy Modeling Toolkit for Python and C++. http://homepages.inf.ed.ac.uk/s0450736/maxent_toolkit.html

Zhao, Bing, and Stephan Vogel. 2002. Adaptive Parallel Sentences Mining from Web Bilingual News Collection. In *Proceedings of IEEE international conference on data mining*, pages 745-750.

Enhancing Cross Document Coreference of Web Documents with Context Similarity and Very Large Scale Text Categorization

Jian Huang
Information Sciences and Technology
Pennsylvania State University
jhuang@ist.psu.edu

Pucktada Treeratpituk
Information Sciences and Technology
Pennsylvania State University
pxt162@ist.psu.edu

Sarah M. Taylor
Lockheed Martin IS&GS
sarah.m.taylor@lmco.com

C. Lee Giles
Information Sciences and Technology
Pennsylvania State University
giles@ist.psu.edu

Abstract

Cross Document Coreference (CDC) is the task of constructing the coreference chain for mentions of a person across a set of documents. This work offers a holistic view of using document-level categories, sub-document level context and extracted entities and relations for the CDC task. We train a categorization component with an efficient flat algorithm using thousands of ODP categories and over a million web documents. We propose to use ranked categories as coreference information, particularly suitable for web documents that are widely different in style and content. An ensemble composite coreference function, amenable to inactive features, combines these three levels of evidence for disambiguation.

A thorough feature importance study is conducted to analyze how these three components contribute to the coreference results. The overall solution is evaluated using the WePS benchmark data and demonstrate superior performance.

1 Introduction

Cross Document Coreference (CDC) is the task to determine whether Named Entities (NE) from different documents refer to the same underlying identity. CDC enables a range of advanced NLP applications such as automated text summarization and question answering (e.g. list-type questions). CDC has mainly been developed from two perspectives.

First, in the Message Understanding Conference (MUC-6), CDC was viewed as an advanced task performed based on a set of Information Extraction (IE) artifacts. IE has been one of the central topics in NLP since the 1970s and gained much success in transforming natural language text to structured text. IE on the Web, however, is inherently very challenging. For one, the Web is comprised of such heterogenous content that IE systems, many of which are developed on tidy and domain-specific corpora, may achieve relatively limited coverage. Also, the content of web documents may not even be in the natural language form. Hence, though IE based features are quite precise, it is rather difficult to achieve good coverage that's necessary to disambiguate person entities on the Web.

Recently, there is significant research interest in a related task called Web Person Search (WePS) (Artiles et al., 2007), which seeks to determine whether two documents refer to the same person given a person name search query. Many systems employed the simple vector space model and word co-occurrence features for this task. Though more robust with better coverage, these methods are more susceptible to irrelevant words with regard to the entity of interest.

Rather than relying solely on IE based or word co-occurrence features, this work adopts a holistic view of the different types of features useful for cross document coreference. Specifically, the main features of our proposed CDC approach are:

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 483–491,
Beijing, August 2010

- The proposed approach covers the entire spectrum of document level, sub-document context level and entity/relation level disambiguation evidence. In particular, we propose to use document categories as robust document level evidence. This comprehensive design naturally combines state-of-the-art categorization, information extraction and IE-driven IR methods and compensates the limitation of each of them.
- The features used in this work are domain independent and thus are particularly suitable for coreferencing web documents.
- The composite pairwise coreference function in this work can readily incorporate a set of heterogenous features that are not always active or are in different ranges, making it easily extensible to additional features. Moreover, we thoroughly study the contribution of each component and its features to gain insight on improving cross document coreference performance.

In this work, three components specialize in generating the aforementioned three levels of features as coreference decisions. Thus we refer to them as *experts*. After reviewing prior work on CDC, we describe the methods of each of these components in detail and present empirical results where appropriate. We then show how these components (and its features) are aggregated to predict pairwise coreference using an ensemble method. We evaluate the contribution of each component and the overall CDC results on a benchmark dataset. Finally, we conclude and discuss future work.

2 Related Work

Compared to the traditional (within-document) coreference resolution problem, cross document coreference is a much harder problem due to the divergence of contents and the lack of consistent discourse information across documents.

(Bagga and Baldwin, 1998b) presented one of the first CDC systems, which relied solely on the contextual words of the named entities. (Gooi and Allan, 2004) used a 55-word window as the context without significant accuracy penalty. As these approaches only considered word co-occurrence, they were more susceptible to genre differences. Recent CDC work has sought Information Extraction (IE) support. Extracted NEs and relationships were considered in (Niu et al., 2004) for improved CDC performance.

Many of these earlier CDC methods were evaluated on small and tidy news articles. CDC for Web documents is even more challenging. (Wan et al., 2005) proposed a web person resolution system called WebHawk, which extracted several attributes such as title, organization, email and phone number using patterns. These features however only covered small amount of disambiguation evidence and certain types of web pages (such as personal home pages). The more recent Web Person Search (WePS) task (Artiles et al., 2007) has created a benchmark dataset which is also used in this work. Different from CDC which aims to resolve mention level NEs, WePS distinguishes *documents* retrieved by a name search query according to the underlying identity. The top-performing system (Chen and Martin, 2007) in this task extracted phrasal contextual and document-level entities as rich features for coreference. Similar IR features are also used by other WePS systems as they are more robust to the variety of web pages (Artiles et al., 2007).

Instead of focusing on local information, (Li et al., 2004) proposed a generative model of entity co-occurrence to capture global document level information. However, inference in generative models is expensive for large scale web data. Our work instead considers document categories/topics that can be efficiently predicted and easily interpretable by users. Hand-tuned weights were used in (Baron and Freedman, 2008) and a linear classifier was used in (Li et al., 2004) to combine the extracted features. Our composite pairwise coreference function is based on an ensemble classifier and is more robust and capable of handling inactive features.

3 Text Categorization Aided CDC

Consider the following scenario for motivation. When a user searches for 'Michael Jordan', the official web page of the basketball player

'Michael Jordan'[1] contains mostly his career statistics, whereas the homepage of 'Michael I. Jordan' the professor[2] contains his titles, contact information and advising students. Neither of these pages contain complete natural language sentences that most IE and NLP tools are designed to process. We propose to use document categories (trained from a very large scale and general purpose taxonomy, Open Directory Project (ODP)) as document level features for CDC. In this example, one can easily differentiate these namesakes by categorizing the former as 'Top/Sports/Basketball/Professional' and the latter as 'Top/Computer/Artificial Intelligence/Machine Learning'. We first introduce the method to categorize Web documents; then we show how to combine these categories for coreferencing.

3.1 Very Large Scale Text Categorization

To handle the web CDC problem, the catagorization component needs to be able to categorize documents of widely different topics. The Open Directory Project (ODP), the largest and most comprehensive human edited directory of the Web[3], contains hundreds of thousands of categories labeled for 2 million Web pages. Leveraging this vast amount of web data and the large Web taxonomy has called for the development of very efficient text categorization methods. There is significant research interest in scaling up to categorize millions of pages to thousands of categories and beyond, called the many class classification setting (Madani and Huang, 2008). Flat classification methods (e.g. (Crammer et al., 2006; Madani and Huang, 2008)), which treat hierarchical categories as flat classes, have been very successful due to their superior scalability and simplicity compared to classical hierarchical one-against-rest categorization. Flat methods also achieve high accuracy that is on par with, or better than, the traditional counterparts.

We adopt a flat multiclass online classification algorithm Passive Aggressive (PA) (Crammer et al., 2006) to predict ranked categories for web documents. For a categorization problem with C categories, PA associates each category k with a weight vector $\mathbf{w}^k$, called its *prototype*. The degree of confidence for predicting category k with respect to an instance $\mathbf{x}$[4] (both in online training and testing) is determined by the similarity between the instance and the prototype — the inner product $\mathbf{w}^k \cdot \mathbf{x}$. PA predicts a ranked list of categories according to this confidence.

PA is a family of online and large-margin based classifiers. Given an instance $(\mathbf{x}_t, y_t)$ during online learning, the multiclass margin $marg$ in PA[5] is the difference between the score of the true category y_t and that of the highest ranked false positive category s, i.e.

$$marg = \mathbf{w}^{y_t} \cdot \mathbf{x_t} - \mathbf{w}^s \cdot \mathbf{x_t} \quad (1)$$

where $s = \arg\max_{s \neq y_t} \mathbf{w}^s \cdot \mathbf{x_t}$.

A positive margin value indicates that the algorithm makes a correct prediction. One is however not only satisfied with a positive margin value, but also seeks to achieve a margin value of at least 1. When this is not satisfied, the online algorithm suffers a multiclass hinge loss:

$$\mathcal{L}_{mc}(\mathbf{w}; (\mathbf{x}_t, y_t)) = \begin{cases} 0 & marg \geq 1 \\ 1 - marg & \text{otherwise} \end{cases}$$

where $\mathbf{w} = (\mathbf{w}^1, .., \mathbf{w}^C)$ denotes the concatenation of the C prototypes (into a vector).

In an online learning step, the PA-II variant updates the category prototype with the solution of this constrained optimization problem,

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_t\|^2 + \mathcal{A}\xi^2 \quad (2)$$

$$s.t. \quad \mathcal{L}_{mc}(\mathbf{w}; (\mathbf{x}_t, y_t)) \leq \xi. \quad (3)$$

Essentially, if the margin is met (also implying no misclassification), PA *passively* accepts the current solution. Otherwise, PA *aggressively* learns the new prototype which satisfies the loss constraint and stays as close to the one previously learned as possible. To cope with *label noise*, PA-II introduces a slack variable ξ in the optimization

[1] See *www.nba.com/playerfile/michael_jordan/index.html*

[2] See *www.eecs.berkeley.edu/∼jordan/*

[3] See *http://www.dmoz.org/about.html* for details.

[4] $\mathbf{x}$ is the vector representation of word frequencies of the corresponding document, L_2 normalized.

[5] For brevity of presentation, we consider the *single label* multiclass categorization setting.

for a gentler update, a technique previously employed to derive soft-margin classifiers (Vapnik, 1998). $\mathcal{A}$ is a parameter that controls the *aggressiveness* of the update.

The solution to the above optimization problem amounts to only changing the two prototypes violating the margin in the update step:

$$\mathbf{w}_{t+1}^{y_t} = \mathbf{w}_t^{y_t} + \tau \mathbf{x}_t \qquad \mathbf{w}_{t+1}^{s} = \mathbf{w}_t^{s} - \tau \mathbf{x}_t$$

where $\tau = \frac{\mathcal{L}_{mc}}{\|\mathbf{x}_t\|^2 + \frac{1}{2\mathcal{A}}}$.

To conclude, PA treats the hierarchy as flat categories for multiclass classification. It is similar to Multiclass Perceptron (Crammer and Singer, 2003) but only updates two vectors per iteration and thus is more efficient.

3.2 Categories as Coreference Evidence

Conceptually, the text categorization component can be viewed as a function that maps a document $\mathbf{d}$ to a ranked list of top K categories along with their respective confidence scores, i.e.

$$\phi(\mathbf{d}) = \{< c_1, s_1 >, .., < c_K, s_K >\}$$

We leverage these document categories to measure the pairwise similarity of any two documents, $\text{sim}(\phi(\mathbf{d}^u), \phi(\mathbf{d}^v))$, for entity disambiguation. Given a taxonomy $\mathcal{T}$, we first formally define the *affinity* between a category c and one of its ancestor category c' in $\mathcal{T}$ as:

$$\text{affinity}(c; c') = 1 - \frac{len(c, c')}{depth(\mathcal{T})}$$

where len is the length of the shortest path between the two categories and *depth(T)* denotes the depth of the taxonomy. In other words, affinity is the complementary of the normalized path length between c and its ancestor c'.

Using graph theory terminology, $\text{LCA}(c_1, c_2)$ denote the *lowest common ancestor* of two categories c_1 and c_2 in $\mathcal{T}$. Given two category lists, $\phi(\mathbf{d}^u) = \{< c_1^u, s_1^u >, .., < c_K^u, s_K^u >\}$ and $\phi(\mathbf{d}^v) = \{< c_1^v, s_1^v >, .., < c_K^v, s_K^v >\}$, we use the $LCA(c_i^u, c_j^v)$ of each category pair c_i^u and c_j^v as the basis to measure similarity. Formally, we transform $\phi(\mathbf{d}^u)$ to a $K \times K$ dimensional vector:

$$\vec{\mathbf{v}}(\mathbf{d}^u) = [\text{affinity}(c_i^u; LCA(c_i^u, c_j^v)) \cdot s_i^u]^T \quad (4)$$

where $i, j = 1..K$. In other words, we project $\phi(\mathbf{d}^u)$ into a vector in the space spanned by the LCAs of category pairs. Using the same bases, we can derive $\vec{\mathbf{v}}(\mathbf{d}^v)$ analogically.

With this transformation, $\phi(\mathbf{d}^u)$ and $\phi(\mathbf{d}^v)$ are expressed in the common bases, i.e. their LCAs. Therefore, the similarity between the top K categories of two documents can be measured by the inner product of these two vectors:

$$\text{sim}(\phi(\mathbf{d}^u), \phi(\mathbf{d}^v)) = \vec{\mathbf{v}}(\mathbf{d}^u) \cdot \vec{\mathbf{v}}(\mathbf{d}^v) \quad (5)$$

3.3 Empirical Studies

To handle the diverse topics of Web documents, we leverage the ODP data to train the many class categorization algorithm. The public ODP data contains 361,621 categories and links to over 2 million pages. We crawled the original web pages from these links, which yielded 1.9 million pages (50GB in size). The taxonomy was condensed to depth three[6] and then very rare categories (having less than 5 instances) were discarded. The data set is created with these categories and the vector representation of the term weights of the extracted raw text. This dataset has 1,889,683 instances and 4,891 categories in total. Finally, stratified 80-20 split was performed on this dataset, i.e. 1.5M pages for training and 377K pages for testing.

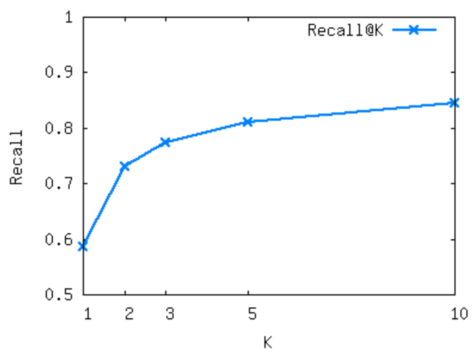


Figure 1: Categorization performance at different positions in the ODP test set.

As we view the taxonomy as a set of flat categories and we are interested in the top K categories, we use the recall at K metric for evaluation. Recall at K is defined as the percentage of instances having their true category ranked

[6]The original taxonomy has average depth 7, which is too deep for the coreference purpose in this work and many categories have too few instances for training.

among the top K slots in the category list. For a single label dataset (most ODP pages have one category) and $K = 1$, this is the accuracy metric in multiclass classification. Note that in the many class setting, recall at 1 is a very strict metric as no credit is given for predicting the parent, children or sibling categories; also, documents may have valid secondary topics not labeled by humans. Figure 1 shows recall at K in the test set. We observe that the algorithm is able to predict the category for 58.7% of the instances in the first rank and more than 77% in top three. There is only diminishing gains when we consider the categories further down the list. Hence we choose to use the similarity of the top 1 and top 3 categories (named `TC1` and `TC3`, respectively) and study their contributions for the CDC task.

3.4 Remarks

In this section, the entire document in the representation of its categories is used as a unit of analysis for CDC. Categorization based CDC works best with namesakes appearing in documents of relatively heterogenous topics, which is usually the case for web documents. Indeed, experienced web searchers would add terms such as 'baseball player' to the name search queries for more relevant results; Wikipedia also (manually) disambiguates namesakes by their professions. Categorization can also be adopted as a robust faceted search system for handling name search queries: users select the interested category/facet to efficiently disambiguate and filter out irrelevant results. The majority of web persons can be readily distinguished by the different underlying categories of the documents where they appear. For more homogeneous corpora or less benevolent cases, the next sections introduce two complementary CDC strategies.

4 Information Extraction for CDC

Consider the following two snippets retrieved with regard to the query 'George Bush':

[Snippet 1]: *"George W. Bush and Bill Clinton are trying to get Congress to allow Haiti to triple the number of exports ..."*

[Snippet 2]: *"George H. W. Bush succeeded Reagan as the 41st U.S. President."*

Using categories alone in this case is insufficient as both will be assigned similar categories such as 'Politics' or 'History/U.S.'. Also, it's not uncommon for these entities to co-occur in the same document and thus making them even more confounding. Properly disambiguating these two mentions requires the usage of local information: for instance, the extraction of full names, the detection of co-occurring NEs and contextual information. We introduce an IE system that extracts precise disambiguation evidence in this section and describe using the extraction context as additional information in the next section.

Our CDC system leverages a state-of-the-art commercial IE system AeroText (Taylor, 2004). The IE system employs manually created knowledge bases with statistically trained models to extract named entities, detect, classify and link relations between NEs. A summary of the most important IE-based features that we use are listed in Table 1. Based on the extracted attributes and relations, we further define their pairwise similarity used as coreference features. This ranges from simple compatibility checking for 'gender', textual soft matching for 'names', to sophisticated semantic matching for 'mentions' and 'locations' using WordNet. (Huang et al., 2009) provides more detailed discussions on the development of these IE based coreference features.

We note that several existing state-of-the-art IE systems are also capable of extracting these features. In particular, Named Entity Recognition (NER) which focuses on a small set of predefined categories of named entities (e.g. persons, organization, location) as well as the detection and tracking of preselected relations have achieved venerable empirical success in practice[7]. Also, within document coreference is a mature and well-studied technology in NLP (e.g. (Ng and Cardie, 2002)). Therefore, our CDC system can readily adopt alternative IE toolkits.

5 Context Matching

As mentioned earlier, achieving high extraction accuracy and coverage for diverse web documents

[7]The Automatic Content Extraction (ACE) evaluation and the Text Analysis Conference (TAC) also have IE-based entity tracking tasks that are relevant to this component.

is still a challenging and open research problem even for the state-of-the-art IE systems. We note that one of the natural outcomes from extraction is the context of the NE of interest, which covers the NE with its surrounding text. For a specific NE, our CDC system uses the context built from the sentences which form the NE's within document coreference chain. The context is then represented as a term vector whose terms are weighted by the TF-IDF weighing scheme. For a pair of NEs, the context matching component measures the cosine similarity of their context term vectors.

Essentially, this component alone is similar to the method presented in the seminal CDC work in (Bagga and Baldwin, 1998b). We however note that simply applying a predetermined threshold on the context similarity for CDC as in this earlier work is not sufficient. First, this method narrowly focuses on the local word occurrence and may miss the *big picture*, i.e. the correlation that exists in the global scope of a document. Also, mere word occurrence is incapable of accounting for the variation of word choices or placing special emphases on evidence such as co-occurring named entities, relations, etc. The categorization and IE components presented earlier in this work overcome these two pitfalls of the simple IR-based approach. We will further showcase the advantage of our comprehensive approach in section 7.2.

6 Composite Pairwise Coreference

In the previous sections, we describe the components to obtain document, sub-document and entity level disambiguation evidence in detail. In this section, we propose to use Random Forest (RF) to combine the experts components into one single composite pairwise similarity score. RF is an ensemble classifier, composed of a collection of randomized decision trees (Breiman, 2001). Each randomized tree is built on a different bootstrap sample of the training data. Randomness is also introduced into the tree construction process: the variable selection for each split is conducted not on the entire feature set, but from a small random subset of features. Gini index is used as the criteria in selecting the best split. Additionally, each tree is unpruned, to keep the prediction bias low. By aggregating many trees that are lowly-correlated (through bootstrap sampling and random variable selection), RF also reduces the prediction variance.

An ensemble method such as Random Forests is very suitable for the CDC task. First, the collection of randomized decision trees is analogous to a panel of different experts, where each makes its decision using different criteria and different features. Previously, RF has been used to aggregate various features in the author disambiguation task (Treeratpituk and Giles, 2009). One of the significant challenges in combining these different features in our CDC setting is that not all of them are always active. For instance, the IE tool may extract an employment relation for one entity and a list relation for another. Also, when the IE tool cannot infer the gender information or when the categorization component does not confidently predict the top K categories (e.g. all with low scores), it's desirable to not supply those features for coreferencing. The traditional technique to impute the missing values, e.g. by replacing them with the mean value, is not suitable in this case. In our work, we specify a special level 'NA' in the decision tree base learner. In our development set, this treatment improves pairwise coreference accuracy by more than 6%.

Figure 2 shows the convergence plot of the composite pairwise coreference function based on Random Forest[8]. We observe that the Out-Of-Bag

[8]The R random forest (Liaw and Wiener, 2002) was used.

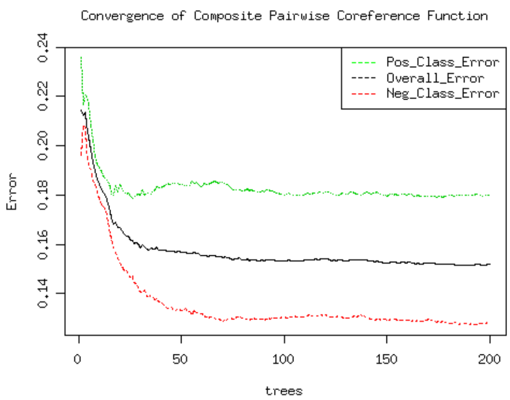


Figure 2: Convergence of OOB errors of the composite pairwise coreference function using the training portion of the WePS dataset.

(OOB) errors [9] drastically decrease with the first 50 trees and then level off (without signs of overfitting). Thus we choose to use the model built with the first 100 trees for prediction. Overall, our model can achieve more than 85% accuracy for pairwise coreference prediction.

7 Experiments

We evaluate our CDC approach with the benchmark dataset from the ACL-2007 SemEval Web Person Search (WePS) evaluation campaign (Artiles et al., 2007). The WePS task is: given a name search query, cluster the search result *documents* according to the underlying referents. Compared to the CDC task which clusters *mention level* entities, a simplifying assumption is made in this task that each document refers to only one identity with respect to the query. The WePS dataset contains the training and test set. The training set contains the top 100 web search results of 49 names from the Web03 corpus (Mann and Yarowsky, 2003), Wikipedia and European Conference on Digital Library (ECDL) participants; the test data are comprised of the top 100 documents of 30 names from Wikipedia, US Census and ACL participants.

Table 1: Expert component and their feature sets.

Feature	Component	Description
`TC1`	Categorization	Sim. of the top 1 categories
`TC3`		Sim. of the top 3 categories
`CNTX`	Context	Sim. of context
`NAME`	IE (attribute)	Sim. of full/first/last names
`MENT`		Sim. of mentions
`GEND`		Sim. of genders
`EMP`	IE (relation)	Sim. of full/first/last names
`LIST`		Sim. of co-occurring persons
`LOC`		Sim. of locations
`FAM`		Sim. of family members

7.1 Evaluation of Pairwise Coreference

We conduct a thorough study of the importance of the individual expert components and their features with the WePS training set. Table 1 shows the three components of the systems, their main features and descriptions.

The importance of these expert components and their features are illustrated in Figure 3. One of the most important features is `CNTX`, this confirms that the prior work on CDC (e.g. (Bagga and Baldwin, 1998b)) can achieve good results with the IE-driven context similarity feature (or its variation). The text categorization component also contributes very important features. In particular, `TC3` is more significant than `TC1` for reducing the Gini index because it recalls more correct categories. On the other hand, `TC1` is slightly more important than `TC3` for its contribution to accuracy, indicating `TC1` is more precise (with less noise categories). For the IE component, attribute features `NAME` and `MENT` are the most useful. As aforementioned, the IE component may not always extract the relation features such as `EMP`, `LIST`, `LOC` and `FAM`, and hence they seemingly have limited effect on model learning (with relatively low reduction in Gini index). These relation features are however very accurate when extracted and are present for prediction. Therefore, they are strong disambiguation evidence and their removal would significantly hamper performance.

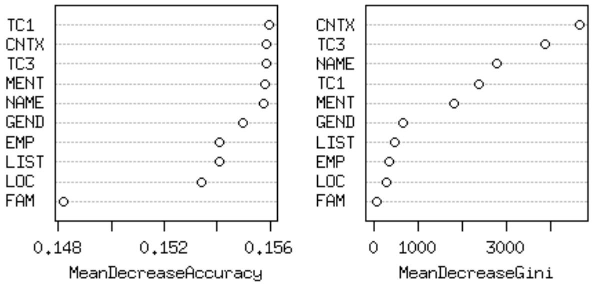


Figure 3: Importance of the expert components and their features found by Random Forest (note the small spread in MeanDecreaseAccuracy).

7.2 Evaluation for Web Person Search

Using the confidence of the pairwise coreference prediction as a distance metric, we adopt a density-based clustering method DBSCAN (Ester et al., 1996) as in (Huang et al., 2006)[10] to induce the person clusters. The final set of evaluation is based on these person clusters generated for the WePS test set.

Two sets of metrics are used to evaluate the overall system. First, we use the B-CUBED

[9]OOB error is an unbiased estimate of test error in RF (Breiman, 2001), computed as the average misclassification rates of each tree with samples not used for its construction.

[10]DBSCAN is a robust and scalable algorithm suitable for clustering relational data. In interest of space, we refer readers to (Ester et al., 1996) for the original algorithm.

Table 2: Cross document coreference performance (I. Pur. denotes inverse purity).

Method	Purity	I. Pur.	*F*	B-CUBED
`CDC`	0.812	0.796	**0.793**	**0.775**
`CNTX`	0.863	0.601	0.678	0.675
`TC1+3`	0.620	0.776	0.660	0.634
`OIO`	1.000	0.482	0.618	0.618
`AIO`	0.279	1.000	0.389	0.238

scores designed in (Bagga and Baldwin, 1998a) for evaluating cross document coreference performance. Second, we use the purity, inverse purity and their F score as in WePS (Artiles et al., 2007). Purity penalizes placing noise entities in a cluster, while inverse purity penalizes splitting coreferent entities into separate clusters.

Table 2 shows the performance of the macro-averaged cross document coreference performance on the WePS test sets. Note that though our evaluation is based on the mention level entities, the baselines One-In-One (`OIO`, placing each entity in a separate cluster) and All-In-One (`AIO`, putting all entities in one cluster) have almost identical results as those in the evaluation[11]. `OIO` can yield good performance, indicating that the names in test data are highly ambiguous. As alluded to in the title, context and categories both are very useful disambiguation features. `CNTX` is essentially very similar to the system presented in (Bagga and Baldwin, 1998b) and is a strong baseline[12] (outperforming $3/4$ of the systems in WePS). Note that `CNTX` has high purity but inferior inverse purity, indicating that using the context extracted by the IE system alone is unable to link many coreferent entities. Interestingly, we observe that using only the top-K categories (`TC1+3`) can also achieve competitive F score, though in a very different manner. `TC1+3` recalls much more coreferent entities (significantly improving inverse purity), but at the same time also introduces noise.

Finally, adding document categories and using IE results (i.e. using all features in Table 1), our CDC system achieves 22% and 18% relative improvement compared to `CNTX` in F (purity) and B-CUBED scores, respectively. In particular, inverse purity improves by 46% relatively, implying that the additional evidence significantly improves the recall of coreferent entities (when there is a lack of context similarity in the traditional method). Overall, the comprehensive approach in this work outperforms the top-tiered systems in the WePS evaluation.

[11]Most person names in this set have only one underlying identity per document; thus the results are comparable despite the simplifying assumption of the WePS evaluation.

[12]We use context similarity 0.2 as the clustering threshold (which has the best performance in training data).

8 Conclusion and Future Work

This work proposes a synergy of three levels of analysis for the web cross document coreference task. On the document level, we use text categories, trained from thousands of ODP categories and over a million pages, as a concise representation of the documents. Categorization is a robust strategy for coreferencing web documents with diverse topics, formats and when there is a lack of extraction coverage or word matching. Two types of sub-document level evidence are also used in our approach. First, we apply an information extraction system to extract attributes and relations of named entities from the documents and perform within document coreference. Second, we use the context of the entities, a natural outcome of the IE system as a focused description of the named entity that may miss the extraction process. A CDC system has been implemented based on the IE and the text categorization components to provide a comprehensive solution to the web CDC task. We demonstrate the importance of each component in our system and benchmark our system with the WePS dataset which shows superior CDC performance.

There are a number of interesting directions for future research. Recently, Open IE was proposed in (Etzioni et al., 2008) for Web information extraction. This can be a more powerful alternative to traditional IE toolkits for Web CDC, though measuring the semantic similarity for a vast variety of relations can be another research issue. Employing external background knowledge such as Wikipedia (Han and Zhao, 2009) while maintaining scalability can also be an orthogonal direction for further improvement.

References

Artiles, Javier, Julio Gonzalo, and Satoshi Sekine. 2007. The SemEval-2007 WePS evaluation: Establishing a benchmark for the web people search task. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval)*, pages 64–69.

Bagga, Amit and Breck Baldwin. 1998a. Algorithms for scoring coreference chains. In *First International Conference on Language Resources and Evaluation Workshop on Linguistics Coreference*.

Bagga, Amit and Breck Baldwin. 1998b. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 36th ACL and 17th COLING*, pages 79–85.

Baron, Alex and Marjorie Freedman. 2008. Who is who and what is what: experiments in cross-document co-reference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 274–283.

Breiman, Leo. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Chen, Ying and James Martin. 2007. Towards robust unsupervised personal name disambiguation. In *Proc. of EMNLP and CoNLL*, pages 190–198.

Crammer, Koby and Yoram Singer. 2003. A family of additive online algorithms for category ranking. *J. Machine Learning Research*, 3:1025–1058.

Crammer, Koby, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research (JMLR)*, 7:551–585.

Ester, M., H. Kriegel, J. Sander, and X. Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd KDD Conference*, pages 226 – 231.

Etzioni, Oren, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the web. *Communications of ACM*, 51(12):68–74.

Gooi, Chung H. and James Allan. 2004. Cross-document coreference on a large scale corpus. In *Proceedings of HLT-NAACL 2004*, pages 9–16.

Han, Xianpei and Jun Zhao. 2009. Named entity disambiguation by leveraging Wikipedia semantic knowledge. In *Proceedings of the 18th Conf. on Information and knowledge management (CIKM)*, pages 215–224.

Huang, Jian, Seyda Ertekin, and C. Lee Giles. 2006. Efficient name disambiguation for large scale databases. In *Proc. of 10th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, pages 536 – 544.

Huang, Jian, Sarah M. Taylor, Jonathan L. Smith, Konstantinos A. Fotiadis, and C. Lee Giles. 2009. Profile based cross-document coreference using kernelized soft relational clustering. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 414–422.

Li, Xin, Paul Morie, and Dan Roth. 2004. Robust reading: Identification and tracing of ambiguous names. In *Proceedings of the Human Language Technology Conference and the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 17–24.

Liaw, Andy and Matthew Wiener. 2002. Classification and regression by randomforest. *R News*, 2(3).

Madani, Omid and Jian Huang. 2008. On updates that constrain the features' connections during learning. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pages 515–523.

Mann, Gideon S. and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Proceedings of the seventh conference on Natural language learning (CoNLL)*, pages 33–40.

Ng, Vincent and Claire Cardie. 2002. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING)*, pages 1–7.

Niu, Cheng, Wei Li, and Rohini K. Srihari. 2004. Weakly supervised learning for cross-document person name disambiguation supported by information extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics (ACL)*, pages 598–605.

Taylor, Sarah M. 2004. Information extraction tools: Deciphering human language. *IT Professional*, 6(6):28 – 34.

Treeratpituk, Pucktada and C. Lee Giles. 2009. Disambiguating authors in academic publications using random forests. In *Proceedings of the ACM/IEEE Joint Conference on Digital libraries (JCDL)*, pages 39–48.

Vapnik, V. 1998. *Statistical Learning Theory*. John Wiley and Sons, Inc., New York.

Wan, Xiaojun, Jianfeng Gao, Mu Li, and Binggong Ding. 2005. Person resolution in person search results: WebHawk. In *Proceedings of the 14th ACM International Conference on Information and Knowledge management (CIKM)*, pages 163–170.

Feature-Rich Discriminative Phrase Rescoring for SMT

Fei Huang and **Bing Xiang**

IBM T. J. Watson Research Center

{huangfe, bxiang}@us.ibm.com

Abstract

This paper proposes a new approach to phrase rescoring for statistical machine translation (SMT). A set of novel features capturing the translingual equivalence between a source and a target phrase pair are introduced. These features are combined with linear regression model and neural network to predict the quality score of the phrase translation pair. These phrase scores are used to discriminatively rescore the baseline MT system's phrase library: boost good phrase translations while prune bad ones. This approach not only significantly improves machine translation quality, but also reduces the model size by a considerable margin.

1 Introduction

Statistical Machine Translation (SMT) systems, including phrase-based (Och and Ney 2002; Koehn et. al. 2003), syntax-based (Yamada and Knight 2001; Galley et. al. 2004) or hybrid systems (Chiang 2005; Zollmann and Venugopal 2006), are typically built with bilingual phrase pairs, which are extracted from parallel sentences with word alignment. Due to the noises in the bilingual sentence pairs and errors from automatic word alignment, the extracted phrase pairs may contain errors, such as

- dropping content words
  (the $num countries ,||个:<null>),
- length mismatch
  (along the lines of the || 的:of)
- content irrelevance
  (the next $num years, ||
  水平:level 方面:aspect 所:<null>)

These incorrect phrase pairs compete with correct phrase pairs during the decoding process, and are often selected when their counts are high (if they contain systematic alignment errors) or certain model costs are low (for example, when some source content words are translated into target function words in an incorrect phrase pair, the language model cost of the incorrect pair may be small, making it more likely that the pair will be selected for the final translation). As a result, the translation quality is degraded when these incorrect phrase pairs are selected.

Various approaches have been proposed over the past decade for the purpose of improving the phrase pair quality for SMT. For example, a term weight based model was presented in (Zhao, et al., 2004) to rescore phrase translation pairs. It models the translation probability with similarities between the query (source phrase) and document (target phrase). Significant improvement was obtained in the translation performance. In (Johnson, et al., 2007; Yang and Zheng, 2009), a statistical significance test was used to heavily prune the phrase table and thus achieved higher precision and better MT performance.

In (Deng, et al., 2008), a generic phrase training algorithm was proposed with the focus on phrase extraction. Multiple feature functions are utilized based on information metrics or word alignment. The feature parameters are optimized to directly maximize the end-to-end system performance. Significant improvement was reported for a small MT task. But when the phrase table is large, such as in a large-scale SMT system, the computational cost of tuning with this approach will be high due to many iterations of phrase extraction and re-decoding.

In this paper we attempt to improve the quality of the phrase table using discriminative phrase rescoring method. We develop extensive set of features capturing the equivalence of bilingual

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 492–500,
Beijing, August 2010

phrase pairs. We combine these features using linear and nonlinear models in order to predict the quality of phrase pairs. Finally we boost the score of good phrases while pruning bad phrases. This approach not only significantly improves the translation quality, but also reduces the phrase table size by 16%.

The paper is organized as follows: in section 2 we discuss two regression models for phrase pair quality prediction: linear regression and neural network. In section 3 we introduce the rich set of features. We describe how to obtain the training data for supervised learning of the two models in section 4. Section 5 presents some approaches to discriminative phrase rescoring using these scores, followed by experiments on model regression and machine translation in section 6.

2 Problem Formulation

Our goal is to predict the translation quality of a given bilingual phrase pair based on a set of features capturing their similarities. These features are combined with linear regression model and neural network. The training data for both models are derived from phrase pairs extracted from small amount of parallel sentences with hand alignment and machine alignment. Details are given in section 4.

2.1 Linear regression model

In the linear regression model, the predicted phrase pair quality score is defined as

$$Sco(e,f)=\sum_i \lambda_i f_i(e,f) \qquad (1)$$

where $f_i(e,f)$ is the feature for the phrase pair (*e,f*), as to be defined in section 3. These feature values can be binary (0/1), integers or real values. λ s are the feature weights to be learned from training data. The phrase pair quality score in the training data is defined as the sum of the target phrase's BLEU score (Papineni et. al. 2002) and the source phrase's BLEU score, where the reference translation is obtained from phrase pairs extracted from human alignment. Details about the training data are given in section 4. The linear regression model is trained using a statistical package R[1]. After training, the learned feature weights are applied on a held-out set of phrase pairs with known quality scores to evaluate the model's regression accuracy.

[1] http://www.r-project.org/

2.2 Neural Network model

A feed-forward back-propagation network (Bryson and Ho, 1969) is created with one hidden layer and 20 nodes. During training, the phrase pair features are fed into the network with their quality scores as expected outputs. After certain iterations of training, the neural net's weights are stable and its mean square error on the training set has been significantly reduced. Then the learned network weights are fixed, and are applied to the test phrase pairs for regression accuracy evaluation. We use MatLab™'s neural net toolkit for training and test.

We will compare both models' prediction accuracy in section 6. We would like to know whether the non-linear regression model outperforms linear regression model in terms of score prediction error, and if fewer regression errors correspond to better translation quality.

3 Feature Description

In this section we will describe the features we use to model the equivalence of a bilingual phrase pair (*e,f*). These features are defined on the phrase pair, its compositional units (words and characters), attributes (POS tags, numbers), co-occurrence frequency, length ratio, coverage ratio and alignment pattern.

- **Phrase:** $P_p(f\,|\,e)$, $P_p(e\,|\,f)$

$$P_p(e\,|\,f)=\frac{C(e,f)}{C(f)} \qquad (2)$$

where $C(e,f)$ is the co-occurrence frequency of the phrase pair (*e,f*), and C(*f*) is the occurrence frequency of the source phrase *f*. $P_p(f\,|\,e)$ is defined similarly.

- Word : $P_w(f\,|\,e)$, $P_w(e\,|\,f)$

$$P_w(e\,|\,f)=\prod_i \max{}_j\, t(e_i\,|\,f_j) \qquad (3)$$

where $t(e_i\,|\,f_j)$ is the lexical translation probability. This is similar to the word-level phrase translation probability, as typically calculated in SMT systems (Brown et. al. 1993). Here we use max instead of sum. $P_w(f \mid e)$ is calculated similarly.

- Character: $P_c(f \mid e), P_c(e \mid f)$

When the source or target words are composed of smaller units, such as characters for Chinese words, or prefix/stem/suffix for Arabic words, we can calculate their translation probability on the sub-unit level. This is helpful for languages where the meaning of a word is closely related to its compositional units, such as Chinese and Arabic.

$$P_c(e \mid f) = \prod_i \max_n t(e_i \mid c_n) \qquad (4)$$

where c_n is the n-th character in the source phrase f (n=1,…,N).

- POS tag: $P_t(f \mid e), P_t(e \mid f)$

In addition to the probabilities estimated at the character, word and phrase levels based on the surface forms, we also compute the POS-based phrase translation probabilities. For each source and target word in a phrase pair, we automatically label their POS tags. Then POS-based probabilities are computed in a way similar to the calculation of the word-level phrase translation probability (formula 3). It is believed that such syntactic information can help to distinguish good phrase pairs from bad ones (for example, when a verb is aligned to a noun, its POS translation probability should be low).

- Length ratio

This feature computes the ratio of the number of content words in the source and target phrases. It is designed to penalize phrases where content words in the source phrase are dropped in the target phrase (or vice versa). The ratio is defined to be 10 if the target phrase has zero content word while the source phrase has non-zero content words. If neither phrase contains a content word, the ratio is defined to be 1.

- Log frequency

This feature takes the logarithm of the co-occurrence frequency of the phrase pair. High frequency phrase pairs are more likely to be correct translations if they are not due to systematic alignment errors.

- Coverage ratio

We propose this novel feature based on the observation that if a phrase pair is a correct translation, it often includes correct sub-phrase pair translations (*decomposition*). Similarly a correct phrase pair will also appear in correct longer phrase pair translations (*composition*) unless it is a very long phrase pair itself. Formally we define the coverage ratio of a phrase pair (*e*,*f*) as:

$$Cov(e,f) = Cov_d(e,f) + Cov_c(e,f) . \ (5)$$

Here $Cov_d(e,f)$ is the decomposition coverage:

$$Cov_d(e,f) = \sum_{f_i \subseteq f} \frac{\sum_{(e_i,f_i) \in P_L} \Delta(e_i, e)}{\sum_{(*,f_i) \in P_L} 1} \qquad (6)$$

where f_i is a sub-phrase of f, and (e_i, f_i) is a phrase pair in the MT system's bilingual phrase library P_L . $\Delta(e_1, e_2)$ is defined to be 1 if $e_1 \subseteq e_2$, otherwise it is 0. For each source sub-phrase f_i , this formula calculates the ratio that its target translation e_i is also a sub-phrase of the target phrase *e*, then the ratio is summed over all the source sub-phrases.

Similarly the composition coverage is defined as

$$Cov_c(e,f) = \sum_{f \subseteq f^j} \frac{\sum_{(e^j,f^j) \in P_L} \Delta(e, e^j)}{\sum_{(*,f^j) \in P_L} 1} \qquad (7)$$

where f^j is any source phrase containing f and e^j is one of f^j's translations in P_L. We call f^j a super-phrase of *f*. For each source super-phrase f^j, this formula calculates the ratio that its target translation e^j is also a super-phrase of the target phrase *e*, then the ratio is summed over all the source super-phrases.

Short phrase pairs (such as a phrase pair with one source word translating into one target word) have less sub-phrases but more super-phrases (for long phrase pairs, it is the other way around).

Combining the two coverage factors produces balanced coverage ratio, not penalizing too short or too long phrases.

- Number match

During preprocessing of the training data, numbers are mapped into a special token ($num) for better generalization. Typically one number corresponds to one special token. During translation numbers should not be arbitrarily dropped or inserted. Therefore we can check whether the source and target phrases have the right number of $num to be matched. If they are the same the number match feature has value 1, otherwise it is 0.

- Alignment pattern

This feature calculates the number of *unaligned* content words in a given phrase pair, where word alignment is obtained simply based on the maximum lexical translation probability of the source (target) word given all the target (source) words in the phrase pair.

Among the above 13 features, the number match feature is a binary feature, the alignment pattern feature is an integer-value feature, and the rest are real-value features. Also note that most features are positively correlated with the phrase translation quality (the greater the feature value, the more likely it is a correct phrase translation) except the alignment pattern feature, where more unaligned content words corresponds to bad phrase translations.

4 Training Data

The training data for both the linear regression and neural network models are bilingual phrase pairs with the above 13 feature values as well as their expected phrase quality scores. The feature values can be computed according to the description in section 3. The expected translation quality score for the phrase pair (*e*,*f*) is defined as

$$B(e,f) = Bleu(e,e^* \mid f) + Bleu(f,f^* \mid e) \quad (8)$$

where e^* is the human translation of the source phrase *f*, and f^* is the human translation of the target phrase *e*. These human translations are obtained from hand alignment of some parallel sentences.

1. Given hand alignment of some bilingual sentence pairs, extract gold phrase translation pairs.
2. Apply automatic word alignment on the same bilingual sentences, and extract phrase pairs. Note that due to the word alignment errors, the extracted phrase pairs are noisy.
3. For each phrase pair (*e*, *f*) in the noisy phrase table, find whether the source phrase *f* also appears in the gold phrase table as (*e**, *f*). If so, use the corresponding target phrase(s) *e** as reference translation(s) to evaluate the BLEU score of the target phrase *e* in the noisy phrase table.
4. Similarly, for each *e* in (*e*, *f*), identify (*e*, *f**) in the gold phrase table and compute the BLEU score of *f* using *f** as the reference.
5. The sum of the above two BLEU scores is the phrase pair's translation quality score.

5 Phrase Rescoring

Given the bilingual phrase pairs' quality score, there are several ways to use them for statistical machine translation.

5.1 Quality score as a decoder feature

A straightforward way is to use the quality scores as an additional feature in the SMT system, combined with other features (phrase scores, word scores, distortion scores, LM scores etc.) for MT hypotheses scoring. The feature weight can be empirically learned using manual tuning or automatic tuning such as MERT (Och 2003). In this situation, all the phrase pairs and their quality scores are stored in the MT system, which is different from the following approach where incorrect phrase translations are pruned.

5.2 Discriminative phrase rescoring

Another approach is to select good and bad phrase pairs based on their predicted quality scores, then discriminatively rescore the phrase pairs in the baseline phrase library. We sort the phrase pairs based on their quality scores in a decreasing order. The bottom *N* phrase pairs are considered as incorrect translations and pruned from the phrase library. The top M phrase pairs P_M are considered as good phrases with correct translations. As identifying correct sub-phrase translation requires accurate word alignment within phrase pairs, which is not easy to obtain due to the lack of rich context information within the phrase pair, we only boost the good phrase pairs' super-phrases in the phrase library. Given a phrase pair (e,f) with phrase co-occurrence count $C(e,f)$, the weighted co-occurrence count is defined as:

$$C'(e,f) = C(e,f) \prod_{(e_i,f_i)\in(e,f)} b_i \qquad (9)$$

where (e_i, f_i) is a *good* sub-phrase pair of (e,f) belonging to P_M, with quality score b_i. Note that if (e,f) contains multiple good sub-phrase pairs, its co-occurrence count will be boosted multiple times. Here the boost factor is defined as the product of quality scores of good sub-phrase pairs. Instead of product, one can also use sum, which did not perform as well in our experiments. The weighted co-occurrence count is used to calculate the new phrase translation scores:

$$P'(e\,|\,f) = \frac{C'(e,f)}{\sum C'(*,f)} \qquad (10)$$

$$P'(f\,|\,e) = \frac{C'(e,f)}{\sum C'(e,*)} \qquad (11)$$

which replace the original phrase translation scores in the SMT system. In addition to phrase co-occurrence count rescoring, the quality scores can also be used to rescore word translation lexicons by updating word co-occurrence counts accordingly.

6 Experiments

We conducted several experiments to evaluate the proposed phrase rescoring approach. First we evaluate the two regression models' quality score prediction accuracy. Secondly, we apply the predicted phrase scores on machine translation tasks. We will measure the improvement on translation quality as well as the reduction of model size. Our experiments are on English-Chinese translation.

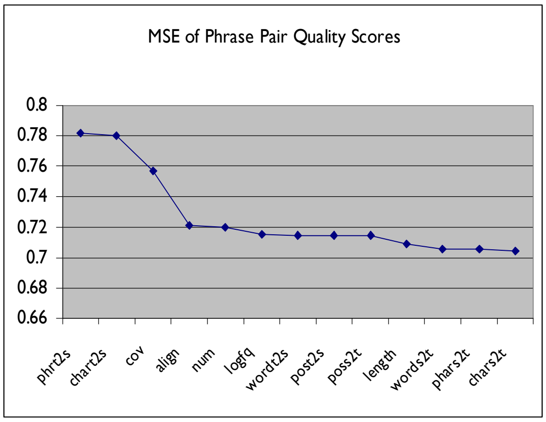


Figure 1. Linear regression model phrase pair prediction MSE curve. Errors are significantly reduced when more features are introduced (phrs2t /phrt2s: phrase source-to-target/target-to-source features; words2t/wordt2s: word-level; chars2t/chart2s: character-level; poss2t/post2s: POS-level; cov: coverage ratio; align: alignment pattern; logfq: log frequency; num: number match; length: length ratio).

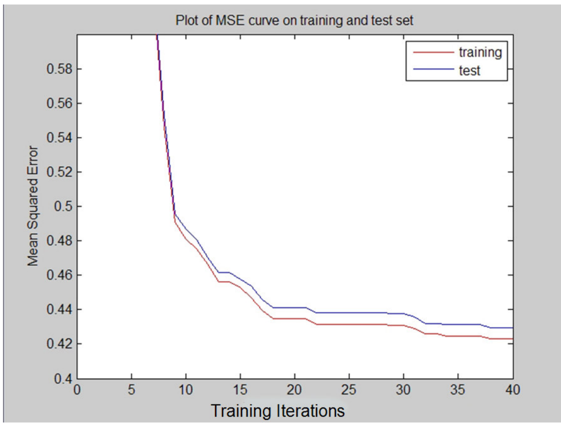


Figure 2. Neural network model phrase pair prediction MSE curve. Errors are significantly reduced with more training iterations.

6.1 Regression model evaluation

We select 10K English-Chinese sentence pairs with both hand alignment and automatic HMM alignment, and extract 106K phrase pairs with true phrase translation quality scores as computed according to formula 8. We choose 53K phrase pairs for regression model training and another 53K phrase pairs for model evaluation. There are 14 parameters to be learned (13 feature weights plus an intercept parameter) for the linear regression model, and 280 weights (13×20

<table>
<tr><th></th><th>Linear Regression</th><th>Neural Network</th></tr>
<tr><td>Good phrase pairs</td><td>and|和|5.52327
amount|金额 数量|4.03006
us|, 美 -|3.91992
her husband|她 丈夫|3.85536
the program|节目 , 一|3.81078
the job|了 这 份 工作|3.77406
shrine|; 靖国神社|3.74336
of course ,|, 当然 , 就 是|3.7174
is only|只 能 是 这|3.69426
visit|访问 只|3.67256
facilities and|设施 , 并 在|3.65402</td><td>rights|权利 |6.96817
has become|已 成为 |4.16468
why|为甚么 |3.82629
by armed|受 武装 |3.62988
o|O |3.47795
of drama|在 戏剧 |3.36601
government and|政府 及 |3.27347
introduction|引进 |3.19113
heart disease|心脏 疾病 |3.11829
heads|首脑们 |3.05467
american consumers|美国 消费者 |2.99706</td></tr>
<tr><td>Bad phrase pairs</td><td>as well|及 其|1.03234
closed|落下 帷幕|1.01271
she was|梅克尔|0.99011
way|改为 双程|0.955918
of a|出 一 种|0.914717
knowledge|察觉|0.875116
made|出席 "|0.837358
the|保持 联络|0.801142
end|之前|0.769938
held|而 进行 的|0.742588</td><td>letter|致函 贵会 |0.39203
, though|尽管 它 |0.37020
levels of|各 级 落实 |0.34892
- board|面板 |0.32826
number of|批 举报 |0.30499
indonesia|苏马尔佐托 |0.27827
xinhua at|$num |0.24433
provinces|安徽 |0.20281
new .|新鲜 之 处 的 , |0.15430
can|的 不同 |0.09502</td></tr>
</table>

Table 2. Examples of good and bad phrase pairs based on the linear regression model and neural network's predicted quality scores.

for the input weight matrix plus 20×1 for the output weight vector) for the neural network model. In both cases, the training data size is much more than the parameters size, so there is no data sparseness problem.

After the model parameters are learned from the training data, we apply the regression model to the evaluation data set, then compute the phrase quality score prediction mean squared error (MSE, also known as the average residual sum of squares):

$$MSE = \frac{1}{K} \sum_k \left[B_p(e_k, f_k) - B_t(e_k, f_k) \right]^2 \quad (12)$$

where B_p is the predicted quality score of the phrase pair (e_k, f_k), while B_t is the true score calculated based on human translations.

Figure 1 shows the reduction of the regression error in the linear regression model trained with different features. One may find that the MSE is significantly reduced (from 0.78 to 0.70) when additional features are added into the regression model.

Similarly, the neural network's MSE curve is shown in Figure 2. It can be seen that the MSE is significantly reduced with more iterations of training (from the initial error of 1.33 to 0.42 after 40 iterations).

Table 2 shows some phrase pairs with high/low quality scores predicted by the linear regression model and the neural network. One can see that both models assign high scores to good phrase translations and low scores to noisy phrase pairs. Although the values of these scores are beyond the range of [0, 2] as defined in formula 8, this is not a problem for our MT tasks, since they are only used as phrase boosting weights or pruning threshold.

6.2 Machine translation evaluation

We test the above phrase rescoring approach on English-Chinese machine translation. The SMT system is a phrase-based decoder similar to the description in (Tillman 2006), where various features are combined within the log-linear framework. These features include source-to-target phrase translation score based on relative frequency, source-to-target and target-to-source word-to-word translation scores, language model score, distortion model scores and word count. The training data for these features are 10M Chi-

	BLEU	NIST	Phrase Table Size
Baseline	38.67	9.3738	3.65M
LR-mtfeat	39.31	9.5356	3.65M
LR-boost (top30k)	39.36	9.5465	3.65M
LR-prune (tail600k)	39.06	9.4890	3.05M
LR-disc (top30K/tail600K)	39.75	9.6388	3.05M
NN-disc (top30K/tail600K)	39.76	9.6547	3.05M
LR-disc tuning	**39.87**	**9.6594**	**3.05M**
Significance-prune	38.96	9.3953	3.01M
Count-Prune	38.65	9.3549	3.05M

Table 3. Translation quality improvements with rescored phrase tables. Best result (1.2 BLEU gain) is obtained with discriminative rescoring by boosting top 30K phrase pairs and pruning bottom 600K phrase pairs, with some weight tuning.

nese-English sentence pairs, mostly newswire and UN corpora released by LDC. The parallel sentences have word alignment automatically generated with HMM and MaxEnt word aligner. Bilingual phrase translations are extracted from these word-aligned parallel corpora. Due to the noise in the bilingual sentence pairs and automatic word alignment errors, the phrase translation library contains many incorrect phrase translations, which lead to inaccurate translations, as seen in Figure 3.

Our evaluation data is NIST MT08 English-Chinese evaluation testset, which includes 1859 sentences from 129 news documents. The automatic metrics are BLEU and NIST scores, as used in the NIST 2008 English-Chinese MT evaluation. Note that as there is no whitespace as Chinese word boundary, the Chinese translations are segmented into characters before scoring in order to reduce the variance and errors caused by automatic word segmentation, which is also done in the NIST MT evaluation.

Table 3 shows the automatic MT scores using the baseline phrase table and rescored phrase tables. When the phrase quality scores from the linear regression model are used as a separate feature in the SMT system (***LR-mtfeat*** as described in section 5.1), the improvement is 0.7 BLEU points (0.16 in terms of NIST scores). By boosting the good phrase pairs (top 30K[2] phrase pairs, ***LR-boost***) from linear regression model, the MT quality is improved by 0.7 BLEU points over the baseline system. Pruning the bad phrase pairs (tail 600K phrase pairs) without using the quality scores as features (***LR-prune***) also improves the MT by 0.4 BLEU points. Combining ***LR-boost*** and ***LR_prune***, a discriminatively rescored phrase table (***LR-disc***) improved the BLEU score by 1.1 BLEU points, and reduce the phrase table size by 16% (from 3.6M to 3.0M phrase pairs). Manually tuning the boosting weights of good phrase pairs leads to additional improvement. Discriminative rescoring using the neural net work scores (***NN-disc***) produced similar improvement.

We also experiment with phrase table pruning using Fisher significant test, as proposed in (Johnson et. al. 2007). We tuned the pruning threshold for the best result. It shows that the significance pruning improves over the baseline by 0.3 BLEU pts with 17.5% reduction in phrase table, but is not as good as our proposed phrase rescoring method. In addition, we also show the MT result using a count pruning phrase table (Count-Prune) where 600K phrase translation pairs are pruned based on their co-occurrence counts. The MT performance of such phrase table pruning is slightly worse than the baseline MT system, and significantly worse than the result using the proposed rescored phrase table.

When comparing the linear regression and neural network models, we find rescoring with both models lead to similar MT improvements, even though the neural network model has much fewer regression errors (0.44 vs. 0.7 in terms of MSE). This is due to the rich parameter space of the neural network.

Overall, the discriminative phrase rescoring improves the SMT quality by 1.2 BLEU points and reduces the phrase table size by 16%. With statistical significance test (Zhang and Vogel 2004), all the improvements are statistically significant with p-value < 0.0001.

Figure 3 presents some English sentences, with phrase translation pairs selected in the final translations (the top one is from the baseline MT system and the bottom one is from the LR-disc system).

[2] These thresholds are empirically chosen.

<table>
<tr><td>Src</td><td>Indonesian bird flu victim contracted virus indirectly:</td></tr>
<tr><td>Baseline</td><td><indonesian bird flu|印尼 禽流感> <virus|病毒> **<victim contracted|感染者>** <indi-rectly :|间接 :></td></tr>
<tr><td>PhrResco</td><td><indonesian bird flu|印尼 禽流感> **<victim|受害者>** **<contracted|感染>** <virus|病毒> <indirectly :|间接 :></td></tr>
<tr><td>Src</td><td>The director of Palestinian human rights group Al-Dhamir, Khalil Abu Shammaleh, said he was also opposed to the move.</td></tr>
<tr><td>Baseline</td><td><the director of|署长 的> <palestinian|巴勒斯坦> <human rights group|人权 团体> **<al -|" 基地 " 组织> <,|,> <abu|Abu>** <khalil|Khalil> <, said he was|表示 , 他> <also opposed to|也 反对> <the move .|这 项 行动 。></td></tr>
<tr><td>PhrResco</td><td><the director of|署长 的> <palestinian|巴勒斯坦> <human rights group|人权 团体> **<al -|al ->** <, khalil|, khalil> **<abu|阿布>** <, said he was|说 , 他> <also opposed to|也 反对> <the move .|这 项 行动 。></td></tr>
<tr><td>Src</td><td>A young female tourist and two of her Kashmiri friends were among the victims.</td></tr>
<tr><td>Baseline</td><td><a young female|有 一 名 年轻 女子> **<tourist and|旅游 和>** <$num of her|她 的 $num 个> <kashmiri|克什米尔> <friends were|网友> <among the|之间 的> <victims .|受害者 。></td></tr>
<tr><td>PhrResco</td><td><a young|一 个 年轻 的> <female|女性> **<tourist and|游客 和>** <$num of her|她 的 $num 个> <kashmiri|克什米尔> <friends were|朋友> <among the|之间 的> <victims .|受害者 。></td></tr>
</table>

Figure 3. Examples of English sentences and their translation, with phrase pairs from baseline system and phrase rescored system. Highlighted text are initial phrase translation errors which are corrected in the PhrResco translations.

We find that incorrect phrase translations in the baseline system (as highlighted with blue bold font) are corrected and better translation results are obtained.

7 Conclusion

We introduced a discriminative phrase rescoring approach, which combined rich features with linear regression and neural network to predict phrase pair translation qualities. Based on these quality scores, we boost good phrase translations while pruning bad phrase translations. This led to statistically significant improvement (1.2 BLEU points) in MT and reduced phrase table size by 16%.

For the future work, we would like to explore other models for quality score prediction, such as SVM. We will want to try other approaches to utilize the phrase pair quality scores, in addition to rescoring the co-occurrence frequency. Finally, we will test this approach in other domain applications and language pairs.

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, Robert L. Mercer. 1993. *The Mathematics of Statistical Machine Translation: Parameter Estimation*, Computational Linguistics, v.19 n.2, June 1993.

Arthur Earl Bryson, Yu-Chi Ho. 1969. *Applied Optimal Control: Optimization, Estimation, and Control*. Blaisdell Publishing Company. p481.

David Chiang. 2005. *A Hierarchical Phrase-based Model for Statistical Machine Translation*. 2005. In Proc. of ACL, pp. 263–270.

Yonggang Deng, Jia Xu, and Yuqing Gao. 2008. *Phrase Table Training for Precision and Recall: What Makes a Good Phrase and a Good Phrase Pair*? In Proc. of ACL/HLT, pp. 81-88.

Michel Galley, Mark Hopkins, Kevin Knight, Daniel Marcu. 2004. *What's in a Translation Rule?* In Proc. of NAACL 2004, pp. 273-280.

Howard Johnson, Joel Martin, George Foster, and Roland Kuhn. 2007. *Improving Translation Quality by Discarding Most of the Phrase Table*. In Proc. of EMNLP-CoNLL, pp. 967-975.

Philipp Koehn, Franz Josef Och, Daniel Marcu. 2003. *Statistical Phrase-based Translation*, In Proc. of NAACL, pp. 48-54.

Franz Josef Och and Hermann Ney. 2003. *A Systematic Comparison of Various Statistical Alignment Models*, Computational Linguistics, v.29 n.1, pp.19-51, March 2003

Franz Josef Och. 2003. *Minimum Error Rate Training in Statistical Machine Translation*, In Proc. of ACL, 2003, pp. 160-167.

Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu. 2002. *BLEU: a Method for Automatic Evaluation of Machine Translation*, In Proc. of ACL, pp. 311-318.

Christoph Tillmann. 2006. *Efficient Dynamic Programming Search Algorithms for Phrase-based SMT*. In Proc. of the Workshop CHPSLP at HLT'06.

Kenji Yamada and Kevin Knight. 2001. *A Syntax-based Statistical Translation Model*, In Proc. of ACL, pp.523-530.

Mei Yang and Jing Zheng. 2009. *Toward Smaller, Faster, and Better Hierarchical Phrase-based SMT*. In Proc. of ACL-IJCNLP, pp. 237-240.

Ying Zhang and Stephan Vogel. 2004. *Measuring Confidence Intervals for the Machine Translation Evaluation Metrics*, In Proc. TMI, pp. 4-6.

Bing Zhao, Stephan Vogel, and Alex Waibel. 2004. *Phrase Pair Rescoring with Term Weighting for Statistical Machine Translation*. In Proc. of EMNLP, pp. 206-213.

Andreas Zollmann and Ashish Venugopal. 2006. *Syntax Augmented Machine Translation via Chart Parsing*. In Proc. of NAACL 2006- Workshop on statistical machine translation.

FactRank: Random Walks on a Web of Facts

Alpa Jain
Yahoo! Labs
alpa@yahoo-inc.com

Patrick Pantel
Microsoft Research
ppantel@microsoft.com

Abstract

Fact collections are mostly built using semi-supervised relation extraction techniques and wisdom of the crowds methods, rendering them inherently noisy. In this paper, we propose to validate the resulting facts by leveraging global constraints inherent in large fact collections, observing that correct facts will tend to match their arguments with other facts more often than with incorrect ones. We model this intuition as a graph-ranking problem over a fact graph and explore novel random walk algorithms. We present an empirical study, over a large set of facts extracted from a 500 million document webcrawl, validating the model and showing that it improves fact quality over state-of-the-art methods.

1 Introduction

Fact bases, such as those contained in *Freebase*, *DBpedia*, *KnowItAll*, and *TextRunner*, are increasingly burgeoning on the Internet, in government, in high tech companies and in academic laboratories. Bar the accurate manual curation typified by Cyc (Lenat, 1995), most fact bases are built using either semi-supervised techniques or wisdom of the crowds techniques, rendering them inherently noisy. This paper describes algorithms to validate and re-rank fact bases leveraging global constraints imposed by the semantic arguments predicated by the relations.

Facts are defined as instances of *n*-ary typed relations such as *acted-in*⟨*movie*, *actor*⟩, *director-of*⟨*movie*, *director*⟩, *born-in*⟨*person*, *date*⟩, and *buy*⟨*person*, *product*, *person*⟩. In all but very small fact bases, relations share an argument type, such as *movie* for the relations *acted-in* and *director-of* in the above example. The hypothesis explored in this paper is that when two fact instances from two relations share the same value for a shared argument type, then the validity of both facts should be increased. Conversely, we also hypothesize that an incorrect fact instance will tend to match a shared argument with other facts far less frequently. For example, consider the following four facts from the relations *acted-in*, *director-of*, and *is-actor*:

t_1: *acted-in*⟨Psycho, Anthony Perkins⟩
t_2: **acted-in*⟨Walt Disney Pictures, Johnny Depp⟩
t_3: *director-of*⟨Psycho, Alfred Hitchcock⟩
t_4: *is-actor*⟨Anthony Perkins⟩

Our confidence in the validity of t_1 increases with the knowledge of t_3 and t_4 since the argument *movie* is shared with t_3 and *actor* with t_4. Similarly, t_1 increases our confidence in the validity of t_3 and t_4. For t_2, we expect to find few facts that will match a *movie* argument with *Walt Disney Pictures*. Facts that share the *actor* argument *Johnny Depp* with t_2 will increase its validity, but the lack of matches on its *movie* argument will decrease its validity.

In this paper, we present *FactRank*, which formalizes the above intuitions by constructing a fact graph and running various random walk graph-ranking algorithms over it to re-rank and validate the facts. A collection of facts is modeled in the form of a graph where nodes are fact instances and edges connect nodes that have the same value for a shared argument type (e.g., t_1 would be linked by an edge to both t_3 and t_4.) Given a graph representation of facts, we explore various random walk algorithms to propagate our confidence in individual facts through the web of facts. We explore algorithms such as PageRank (Page et al., 1999) as well as propose novel algorithms that leverage several unique characteristics of fact graphs. Finally, we present an empirical analysis, over a large collection of facts extracted from a 500 mil-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 501–509,
Beijing, August 2010

lion document webcrawl, supporting our model and confirming that global constraints in a fact base can be leveraged to improve the quality of the facts. Our proposed algorithms are agnostic to the sources of a fact base, however our reported experiments were carried over a state-of-the-art semi-supervised extraction system. In summary, the main contributions of this paper are:

- We formalize the notion of ranking facts in a holistic manner by applying graph-based ranking algorithms (Section 2).
- We propose novel ranking algorithms using random walk models on facts (Section 3).
- We establish the effectiveness of our approach through an extensive experimental evaluation over a real-life dataset and show improvements over state-of-the-art ranking methods (Section 4).

2 Fact Validation Revisited

We denote an n-ary ***relation*** r with typed arguments $t_1, t_2, \cdots, t_n$ as $r\langle t_1, t_2, \cdots t_n\rangle$. In this paper, we limit our focus to unary and binary relations. A ***fact*** is an instance of a relation. For example, *acted-in*⟨*Psycho, Anthony Perkins*⟩ is a fact from the *acted-in*$\langle movie, actor\rangle$ relation.

Definition 2.1 [Fact base]: *A* fact base *is a collection of facts from several relations. Textrunner and Freebase are example fact bases (note that they also contain knowledge beyond facts such as entity lists and ontologies.)* □

Definition 2.2 [Fact farm]: *A* fact farm *is a subset of interconnected relations in a fact base that share arguments among them.* □

For example, consider a fact base consisting of facts for relations involving movies, organizations, products, etc., of which the relations *acted-in* and *director-of* could form a MOVIES fact farm.

Real-world fact bases are built in many ways. Semi-supervised relation extraction methods include *KnowItAll* (Etzioni et al., 2005), *TextRunner* (Banko and Etzioni, 2008), and many others such as (Riloff and Jones, 1999; Pantel and Pennacchiotti, 2006; Paşca et al., 2006; Mintz et al., 2009). Wisdom of the crowds methods include DBpedia (Auer et al., 2008) and Freebase which extracts facts from various open knowledge bases and allow users to add or edit its content.

Most semi-supervised relation extraction methods follow (Hearst, 1992). Starting with a relatively small set of seed facts, these extractors iteratively learn patterns that can be instantiated to identify new facts. To reflect their confidence in an extracted fact, extractors assign an *extraction score* with each fact. Methods differ widely in how they define the *extraction score*. Similarly, many extractors assign a *pattern score* to each discovered pattern. In each iteration, the highest scoring patterns and facts are saved, which are used to seed the next iteration. After a fixed number of iterations or when a termination condition is met, the instantiated facts are ranked by their *extraction score*.

Several methods have been proposed to generate such ranked lists (e.g., (Riloff and Jones, 1999; Banko and Etzioni, 2008; Matuszek et al., 2005; Pantel and Pennacchiotti, 2006; Paşca et al., 2006). In this paper, we re-implement the large-scale state-of-the-art method proposed by Paşca et al. (2006). This pattern learning method generates binary facts and computes the extraction scores of a fact based on (a) the scores of the patterns that generated it, and (b) the distributional similarity score between the fact and the seed facts. We computed the distributional similarity between arguments using (Pantel et al., 2009) over a large crawl of the Web (described in Section 4.1). Other implementation details follow (Paşca et al., 2006).

In our experiments, we observed some interesting ranking problems as illustrated by the following example facts for the *acted-in* relation:

id:	Facts (#Rank)
t_1:	*acted-in*⟨Psycho, Anthony Perkins⟩ (#26)
t_2:	**acted-in*⟨Walt Disney Pictures, Johnny Depp⟩ (#9)

Both t_1 and t_2 share similar contexts in documents (e.g., ⟨*movie*⟩ *film starring* ⟨*actor*⟩ *and* ⟨*movie*⟩ *starring* ⟨*actor*⟩), and this, in turn, boosts the pattern-based component of the extraction scores for t_1. Furthermore, due to the ambiguity of the term *psycho*, the distributional similarity-based component of the scores for fact t_2 is also lower than that for t_1.

Relations	id : Facts
acted-in	t_1 : ⟨Psycho, Anthony Perkins⟩
	t_2 : *⟨Walt Disney Pictures, Johnny Depp⟩
director-of	t_3 : ⟨Psycho, Alfred Hitchcock⟩
producer-of	t_4 : ⟨Psycho, Hilton Green⟩
is-actor	t_5 : ⟨Anthony Perkins⟩
	t_6 : ⟨Johnny Depp⟩
is-director	t_7 : ⟨Alfred Hitchcock⟩
is-movie	t_8 : ⟨Psycho⟩

Table 1: Facts share arguments across relations which can be exploited for validation.

Our work in this paper is motivated by the following observation: the ranked list generated by an individual extractor does not leverage any global information that may be available when considering a fact farm in concert. To understand the information available in a fact farm, consider a MOVIES fact farm consisting of relations, such as, *acted-in*, *director-of*, *producer-of*, *is-actor*, *is-movie*, and *is-director*. Table 1 lists sample facts that were generated in our experiments for these relations[1]. In this example, we observe that for t_1 there exist facts in foreign relations, namely, *director-of* and *producer-of* that share the same value for the *Movie* argument, and intuitively, facts t_3 and t_4 add to the validity of t_1. Furthermore, t_1 shares the same value for the *Actor* argument with t_5. Also, t_3, which is expected to boost the validity of t_1, itself shares values for its arguments with facts t_4 and t_7, which again intuitively adds to the validity of t_1. In contrast to this *web of facts* generated for t_1, the fact t_2 shares only one of its argument value with one other fact, i.e., t_6.

The above example underscores an important observation: *How does the web of facts generated by a fact farm impact the overall validity of a fact?* To address this question, we hypothesize that facts that share arguments with many facts are more reliable than those that share arguments with few facts. To capture this hypothesis, we model a web of facts for a farm using a graph-based representation. Then, using graph analysis algorithms, we propagate reliability to a fact using the scores of other facts that recursively connect to it.

Starting with a fact farm, to validate the facts in each consisting relation, we:

(1) Identify arguments common to relations in the farm.
(2) Run extraction methods to generate each relation.
(3) Construct a graph-based representation of the extracted facts using common arguments identified in Step (1) (see Section 3.1 for details on constructing this graph.)
(4) Perform link analysis using random walk algorithms over the generated graph, propagating scores to each fact through the interconnections (see Section 3.2 for details on various proposed random walk algorithms).
(5) Rank facts in each relation using the scores generated in Step (4) or by combining them with the original extraction scores.

For the rest of the paper, we focus on generating better ranked lists than the original rankings proposed by a state-of-the-art extractor.

3 FactRank: Random Walk on Facts

Our approach considers a fact farm holistically, leveraging the global constraints imposed by the semantic arguments of the facts in the farm. We model this idea by constructing a graph representation of the facts in the farm (Section 3.1) over which we run graph-based ranking algorithms. We give a brief overview of one such ranking algorithm (Section 3.2) and present variations of it for fact re-ranking (Section 3.3). Finally, we incorporate the original ranking from the extractor into the ranking produced by our random walk models (Section 3.4).

3.1 Graph Representation of Facts

Definition 3.1 *We define a* fact graph *FG(V, E), with V nodes and E edges, for a fact farm, as a graph containing facts as nodes and a set of edges between these nodes. An edge between nodes v_i and v_j indicates that the facts share the same value for an argument that is common to the relations that v_i and v_j belong to.* □

Figure 1 shows the fact graph for the example in Table 1 centered around the fact t_1.

Note on the representation: The above graph representation is just one of many possible options. For instance, instead of representing facts by nodes, nodes could represent the arguments of facts (e.g., *Psycho*) and nodes could be connected by edges if they occur together in a fact. The task of studying a “best” representation remains a future work direction. However, we believe that our proposed methods can be easily adapted to other such graph representations.

[1]The *is-actor*⟨actor⟩, *is-director*⟨director⟩, and *is-movie*⟨movie⟩ relations are equivalent to the relation *is-a*⟨c-instance, class⟩ where $class \in \{actor, director, movie\}$.

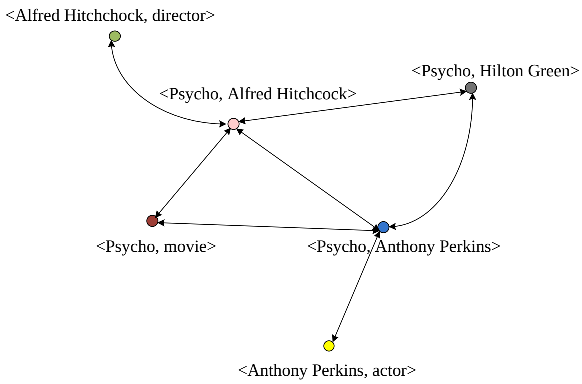


Figure 1: Fact graph centered around t_1 in Table 1.

3.2 The FactRank Hypothesis

We hypothesize that connected facts increase our confidence in those facts. We model this idea by propagating *extraction scores* through the fact graph similarly to how authority is propagated through a hyperlink graph of the Web (used to estimate the importance of a webpage). Several link structure analysis algorithms have been proposed for this goal, of which we explore a particular example, namely, PageRank (Page et al., 1999). The premise behind PageRank is that given the hyperlink structure of the Web, when a page v generates a link to page u, it confers some of its importance to u. Therefore, the importance of a webpage u depends on the number of pages that link to u and furthermore, on the importance of the pages that link to u. More formally, given a directed graph $G = (V, E)$ with V vertices and E edges, let $I(u)$ be the set of nodes that link to a node u and $O(v)$ be the set of nodes linked by v. Then, the importance of a node u is defined as:

$$p(u) = \sum_{v \in I(u)} \frac{p(v)}{|O(v)|} \qquad (1)$$

The PageRank algorithm iteratively updates the scores for each node in G and terminates when a convergence threshold is met. To guarantee the algorithm's convergence, G must be irreducible and aperiodic (i.e., a connected graph). The first constraint can be easily met by converting the adjacency matrix for G into a stochastic matrix (i.e., all rows sum up to 1.) To address the issue of periodicity, Page et al. (1999) suggested the following modification to Equation 1:

$$p(u) = \frac{1-d}{|V|} + d \cdot \sum_{v \in I(u)} \frac{p(v)}{|O(v)|} \qquad (2)$$

where d is a damping factor between 0 and 1, which is commonly set to 0.85. Intuitively, PageRank can be viewed as modeling a "random walker" on the nodes in G and the score of a node, i.e., PageRank, determines the probability of the walker arriving at this node.

While our method makes use of the PageRank algorithm, we can also use other graph analysis algorithms (e.g., HITS (Kleinberg, 1999)). A particularly important property of the PageRank algorithm is that the stationary scores can be computed for *undirected* graphs in the same manner described above, after replacing each undirected edge by a bi-directed edge. Recall that the edges in a fact graph are bi-directional (see Figure 1).

3.3 Random Walk Models

Below, we explore various random walk models to assign scores to each node in a fact graph *FG*.

3.3.1 Model Implementations

Pln: Our first method applies the traditional PageRank model to *FG* and computes the score of a node u using Equation 2.

Traditional PageRank, as is, does not make use of the *strength* of the links or the nodes connected by an edge. Based on this observation, researchers have proposed several variations of the PageRank algorithm in order to solve their problems. For instance, variations of random walk algorithms have been applied to the task of extracting important words from a document (Hassan et al., 2007), for summarizing documents (Erkan and Radev, 2004), and for ordering user preferences (Liu and Yang, 2008). Following the same idea, we build upon the discussion in Section 3.2 and present random walk models that incorporate the strength of an edge.

Dst: One improvement over ***Pln*** is to distinguish between nodes in *FG* using the extraction scores of the facts associated with them: extraction methods such as our reimplementation of (Paşca et al., 2006) assign scores to each output fact to reflect its confidence in it (see Section 3.2). Intuitively, a higher scoring node that connects to u should increase the importance of u more than a connection from a lower scoring node. Let $I(u)$ be the set of nodes that link to u and $O(v)$ be the set of nodes linked by v. Then, if $w(u)$ is the extraction score for the fact represented by node u, the score for node u is defined:

$$p(u) = \frac{1-d}{|V|} + d \cdot \sum_{v \in I(u)} \frac{w(v) \times p(v)}{|O(v)|} \quad (3)$$

where $w(v)$ is the confidence score for the fact represented by v. Naturally, other (externally derived) extraction scores can also be substituted for $w(v)$.

Avg: We can further extend the idea of determining the strength of an edge by combining the extraction scores of *both* nodes connected by an edge. Specifically,

$$p(u) = \frac{1-d}{|V|} + d \cdot \sum_{v \in I(u)} \frac{avg(u,v) \times p(v)}{|O(v)|} \quad (4)$$

where $avg(u,v)$ is the average of the extraction scores assigned to the facts associated with nodes u and v.

Nde: In addition to using extraction scores, we can also derive the strength of a node depending on the number of *distinct* relations it connects to. For instance, in Figure 1, t_1 is linked to four distinct relations, namely, *director-of*, *producer-of*, *is-actor*, *is-movie*, whereas, t_2 is linked to one relation, namely, *is-actor*. We compute $p(u)$ as:

$$p(u) = \frac{1-d}{|V|} + d \cdot \sum_{v \in I(u)} \frac{(\alpha \cdot w(v) + (1-\alpha) \cdot r(v)) \times p(v)}{|O(v)|} \quad (5)$$

where $w(v)$ is the confidence score for node v and $r(v)$ is the fraction of total number of relations in the farm that contain facts with edges to v.

3.3.2 Dangling nodes

In traditional hyperlink graphs for the Web, dangling nodes (i.e., nodes with no associated edges) are considered to be of low importance which is appropriately represented by the scores computed by the PageRank algorithm. However, an important distinction from this setting is that fact graphs are sparse causing them to have valid facts with no counterpart matching arguments in other relation, thus rendering them dangling. This may be due to several reasons, e.g., extractors often suffer from less than perfect recall and they may miss valid facts. In our experiments, about 10% and 40% of nodes from *acted-in* and *director-of*, respectively, were dangling nodes.

Handling dangling nodes in our extraction-based scenario is a particularly challenging issue: while demoting the validity of dangling nodes could critically hurt the quality of the facts, lack of global information prevents us from systematically introducing them into the re-ranked lists. We address this issue by maintaining the original rank positions when re-ranking dangling nodes.

3.4 Incorporating Extractor Ranks

Our proposed random walk ranking methods ignore the ranking information made available by the original relation extractor (e.g., (Paşca et al., 2006) in our implementation). Below, we propose two ways of combining the ranks suggested by the original ranked list O and the re-ranked list G, generated using the algorithms in Section 3.3.

R-Avg: The first combination method computes the average of the ranks obtained from the two lists. Formally, if $O(i)$ is the original rank for fact i and $G(i)$ is the rank for i in the re-ranked list, the combined rank $M(i)$ is computed as:

$$M(i) = \frac{O(i) + G(i)}{2} \quad (6)$$

R-Wgt: The second method uses a weighted average of the ranks from the individual lists:

$$M(i) = \frac{w_o \cdot O(i) + (1-w_o) \cdot G(i)}{2} \quad (7)$$

In practice, this linear combination can be learned; in our experiments, we set them to $w_o = 0.4$ based on our observations over an independent training set. Several other combination functions could also be applied to this task. For instance, we explored the *min and* max functions but observed little improvements.

4 Experimental Evaluation

4.1 Experimental Setup

Extraction method: For our extraction method, we reimplemented the method described in (Paşca et al., 2006) and further added a validation layer on top of it based on Wikipedia (we boosted the scores of a fact if there exists a Wikipedia page for either of the fact's arguments, which mentions the other argument.) This state-of-the-art method forms a *strong* baseline in our experiments.

Corpus and farms: We ran our extractor over a large Web crawl consisting of 500 million English

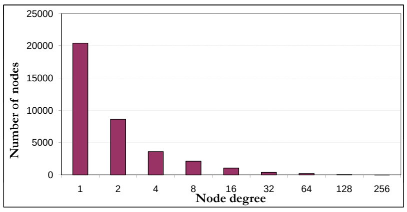


Figure 2: Degree distribution for MOVIES.

webpages crawled by the Yahoo! search engine. We removed paragraphs containing fewer than 50 tokens and then removed all duplicate sentences. The resulting corpus consists of over 5 million sentences. We defined a farm, MOVIES, with relations, *acted-in*, *director-of*, *is-movie*, *is-actor*, and *is-director*.

Evaluation methodology: Using our extraction method over the Web corpus, we generate over 100,000 facts for the above relations. However, to keep our evaluation manageable, we draw a random sample from these facts. Specifically, we first generate a ranked list using the extraction scores output by our extractor. We will refer to this method as ***Org*** (original). We then generate a fact graph over which we will run our methods from Section 3.3 (each of which will re-rank the facts). Figure 2 shows the degree, i.e., number of edges, distribution of the fact graph generated for MOVIES. We ran ***Avg***, ***Dst***, ***Nde***, ***R-Avg***, and ***R-Wgt*** on this fact graph and using the scores we re-rank the facts for each of the relations. In Section 4.2, we will discuss our results for the *acted-in* and *director-of* relations.

Fact Verification: To verify whether a fact is valid or not, we recruit human annotators using the paid service Mechanical Turk. For each fact, two annotations were requested (keeping the total cost under $100). The annotators were instructed to mark incorrect facts as well as disallow any values that were not "well-behaved." For instance, *acted-in*⟨*Godfather, Pacino*⟩ is correct, but *acted-in*⟨*The, Al Pacino*⟩ is incorrect. We manually adjudicated 32% of the facts where the judges disagreed.

Evaluation metrics: Using the annotated facts, we construct a goldset S of facts and compute the precision of a list L as: $\frac{|L \cap S|}{|S|}$. To compare the effectiveness of the ranked lists, we use average precision, a standard measure in information retrieval for evaluating ranking algorithms, defined

Method	Average precision		
	30%	50%	100%
Org	0.51	0.39	0.38
Pln	0.44	0.35	0.32
Avg	0.55	0.44	0.42
Dst	0.54	0.44	0.41
Nde	0.53	0.40	0.41
R-Avg	0.58	0.46	**0.45**
R-Wgt	**0.60**	**0.56**	0.44

Table 2: Average precision for *acted-in* for varying proportion of fact graph of MOVIES.

Method	Average precision		
	30%	50%	100%
Org	0.64	0.69	0.66
Pln	0.69	0.67	0.59
Avg	0.69	0.70	0.64
Dst	0.67	0.69	0.64
Nde	0.69	0.69	0.64
R-Avg	0.70	0.70	0.64
R-Wgt	**0.71**	**0.71**	**0.69**

Table 3: Average precision for *director-of* for varying proportion of fact graph of MOVIES.

as: $A_p(L) = \frac{\sum_{i=1}^{|L|} P(i) \cdot isrel(i)}{\sum_{i=1}^{|L|} isrel(i)}$, where $P(i)$ is the precision of L at rank i, and *isrel(i)* is 1 if the fact at rank i is in S, and 0 otherwise. We also study the precision values at varying ranks in the list. For robustness, we report the results using 10-fold cross validation.

4.2 Experimental Results

Effectiveness of graph-based ranking: Our first experiment studies the overall quality of the ranked lists generated by each method. Table 2 compares the average precision for *acted-in*, with the maximum scores highlighted for each column. We list results for varying proportions of the original fact graph (30%, 50%, and 100%). Due to our small goldset sizes, these results are not statistically significant over ***Org***, however we consistently observed a positive trend similar to those reported in Table 2 over a variety of evaluation sets generated by randomly building 10-folds of all the facts.

Overall, the ***Avg*** method offers a competitive alternative to the original ranked list generated by the extractor ***Org***: not only are the average precision values for ***Avg*** higher than ***Org***, but as we will see later, the rankings generated by our graph-based methods exhibits some positive unique characteristics. These experiments also

R	*Org*	*Pln*	*Avg*	*Dst*	*Nde*	*R-Avg*	*R-Wgt*
5	0.44	0.40	0.52	0.48	0.40	0.52	**0.56**
10	0.36	0.36	**0.42**	0.38	0.36	0.36	0.36
15	0.287	0.24	**0.30**	0.28	0.26	**0.30**	**0.30**
20	0.26	0.26	0.26	0.26	0.26	**0.27**	**0.27**
21	0.27	0.27	0.27	0.27	0.27	0.27	0.27

Table 4: Precision at varying ranks for the *acted-in* relation (R stands for Ranks).

R	*Org*	*Pln*	*Avg*	*Dst*	*Nde*	*R-Avg*	*R-Wgt*
5	0.58	0.68	**0.70**	0.68	0.64	0.66	**0.70**
10	0.60	0.57	0.59	0.58	0.59	0.6	**0.69**
15	0.57	0.53	**0.58**	0.56	0.56	0.56	**0.60**
20	0.57	0.57	0.58	0.58	0.58	0.58	**0.60**
25	**0.60**	0.54	0.56	0.57	0.56	0.57	0.57
30	0.57	0.57	0.57	0.57	0.57	0.58	**0.59**
33	0.56	0.56	0.56	0.56	0.56	0.56	0.56

Table 5: Precision at varying ranks for the *director-of* relation (R stands for Ranks).

confirm our initial observations: using traditional PageRank (***Pln***) is not desirable for the task of re-ranking facts (see Section 3.3). Our modifications to the PageRank algorithm (e.g., ***Avg***, ***Dst***, ***Nde***) consistently outperform the traditional PageRank algorithm (***Pln***). The results also underscore the benefit of combining the original extractor ranks with those generated by our graph-based ranking algorithms with ***R-Wgt*** consistently leading to highest or close to the highest average precision scores.

In Table 3, we show the average precision values for *director-of*. In this case, the summary statistic, average precision, does not show many differences between the methods. To take a finer look into the quality of these rankings, we investigated the precision scores at varying ranks across the methods. Table 4 and Table 5 show the precision at varying ranks for *acted-in* and *director-of* respectively. The maximum precision values for each rank are highlighted.

For *acted-in* again we see that ***Avg***, ***R-Avg***, ***R-Wgt*** outperform ***Org*** and ***Pln*** at *all* ranks, and ***Dst*** outperforms ***Org*** at two ranks. While the method ***Nde*** outperforms ***Org*** for a few cases, we expected it to perform better. Error analysis revealed that the sparsity of our fact graph was the problem. In our MOVIES fact graph, we observed very few nodes that are linked to *all* possible relation types, and the scores used by ***Nde*** rely on being able to identify nodes that link to numerous relation types. This problem can be alleviated

#Relation	*Avg*	*Dst*	*Nde*
2	0.35	0.34	0.33
3	0.35	0.35	0.34
4	0.37	0.36	0.35
5	0.38	0.38	0.37
6	0.42	0.41	0.41

Table 6: Average precision for *acted-in* for varying number of relations in the MOVIES fact farm.

by reducing the sparsity of the fact graphs (e.g., by allowing edges between nodes that are "similar enough"), which we plan to explore as future work. For *director-of*, Table 5 now shows that for small ranks (less than 15), a small (but consistent in our 10-folds) improvement is observed when comparing our random walk algorithms over ***Org***.

While our proposed algorithms show a consistent improvement for *acted-in*, the case of *director-of* needs further discussion. For both average precision and precision vs. rank values, ***Avg***, ***R-Avg***, and ***R-Wgt*** are similar or slightly better than ***Org***. We observed that the graph-based algorithms tend to bring together "clusters" of noisy facts that may be spread out in the original ranked list of facts. To illustrate this point, we show the ten lowest scoring facts for the *director-of* relation. Table 7 shows these ten facts for ***Org*** as well as ***Avg***. These examples highlight the ability of our graph-based algorithms to demote noisy facts.

Effect of number of relations: To understand the effect of the number of relations in a farm (and hence connectivity in a fact graph), we verified the re-ranking quality of our proposed methods on various subsets of the MOVIES fact farm. We generated five different subsets, one with 2 relations, another with 3 relations, and three more with four, five, and six relations (note that although we have 5 relations in the farm, *is-movie* can be used in combination with both *acted-in* and *director-of*, thus yielding six relations to ablate.) Table 6 shows the results for *acted-in*. Overall, performance improves as we introduce more relations (i.e., more connectivity). Once again, we observe that the performance deteriorates for sparse graphs: using very few relations results in degenerating the average precision of the original ranked list. The issue of identifying the "right" characteristics of the fact graph (e.g., number of relations, degree distribution, etc.) remains future work.

Org	*Avg*
⟨david mamet, bob rafelson⟩	⟨ drama, nicholas ray⟩
⟨cinderella, wayne sleep⟩	⟨ drama, mitch teplitsky official⟩
⟨mozartdie zauberflte, julie taymor⟩	⟨ hollywood, marta bautis⟩
⟨matthew gross, julie taymor⟩	⟨ hollywood, marek stacharski⟩
⟨steel magnolias, theater project⟩	⟨ drama, kirk shannon-butts⟩
⟨rosie o'donnell, john badham⟩	⟨ drama, john pietrowski⟩
⟨my brotherkeeper, john badham⟩	⟨ drama, john madden starring⟩
⟨goldie hawn, john badham⟩	⟨ drama, jan svankmajer⟩
⟨miramaxbad santa, terry zwigoff⟩	⟨ drama, frankie sooknanan⟩
⟨premonition, alan rudolph⟩	⟨ drama, dalia hager⟩

Table 7: Sample facts for *director-of* at the bottom of the ranked list generated by (a) ***Org*** and (b) ***Avg***.

Evaluation conclusion: We demonstrated the effectiveness of our graph-based algorithms for re-ranking facts. In general, ***Avg*** outperforms ***Org*** and ***Pln***, and we can further improve the performance by using a combination-based ranking algorithm such as ***R-Wgt***. We also studied the impact of the size of the fact graphs on the quality of the ranked lists and showed that increasing the density of the fact farms improves the ranking using our methods.

5 Related Work

Information extraction from text has received significant attention in the recent years (Cohen and McCallum, 2003). Earlier approaches relied on hand-crafted extraction rules such as (Hearst, 1992), but recent efforts have developed supervised and semi-supervised extraction techniques (Riloff and Jones, 1999; Agichtein and Gravano, 2000; Matuszek et al., 2005; Pantel and Pennacchiotti, 2006; Paşca et al., 2006; Yan et al., 2009) as well as unsupervised techniques (Davidov and Rappoport, 2008; Mintz et al., 2009). Most common methods today use semi-supervised pattern-based learning approaches that follow (Hearst, 1992), as discussed in Section 2. Recent work has also explored extraction-related issues such as, *scalability* (Paşca et al., 2006; Ravichandran and Hovy, 2002; Pantel et al., 2004; Etzioni et al., 2004), *learning extraction schemas* (Cafarella et al., 2007a; Banko et al., 2007), and *organizing extracted facts* (Cafarella et al., 2007b). There is also a lot of work on deriving extraction scores for facts (Agichtein and Gravano, 2000; Downey et al., 2005; Etzioni et al., 2004; Pantel and Pennacchiotti, 2006).

These extraction methods are complementary to our general task of fact re-ranking. Since our proposd re-ranking algorithms are agnostic to the methods of generating the initial facts and since they do not rely on having available corpus statistics, we can use any of the available extractors in combination with any of the scoring methods. In this paper, we used Paşca et al.'s (2006) state-of-the-art extractor to learn a large set of ranked facts.

Graph-based ranking algorithms have been explored for a variety of text-centric tasks. Random walk models have been built for document summarization (Erkan and Radev, 2004), keyword extraction (Hassan et al., 2007), and collaborative filtering (Liu and Yang, 2008). Closest to our work is that of Talukdar et al. (2008) who proposed random walk algorithms for learning instances of semantic classes from unstructured and structured text. The focus of our work is on random walk models over fact graphs in order to re-rank collections of facts.

6 Conclusion

In this paper, we show how information available in a farm of facts can be exploited for re-ranking facts. As a key contribution of the paper, we modeled fact ranking as a graph ranking problem. We proposed random walk models that determine the validity of a fact based on (a) the number of facts that "vote" for it, (b) the validity of the voting facts, and (c) the extractor's confidence in these voting facts. Our experimental results demonstrated the effectiveness of our algorithms, thus establishing a stepping stone towards exploring graph-based frameworks for fact validation. While this paper forms the basis of employing random walk models for fact re-ranking, it also suggests several interesting directions for future work. We use and build upon PageRank, however, several alternative algorithms from the link analysis literature could be adapted for ranking facts. Similarly, we employ a single (simple) graph-based representation that treats all edges the same and exploring richer graphs that distinguish between edges supporting different arguments of a fact remains future work.

References

[Agichtein and Gravano2000] Agichtein, Eugene and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *DL-00*.

[Auer et al.2008] Auer, S., C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. 2008. Dbpedia: A nucleus for a web of open data. In *ISWC+ASWC 2007*.

[Banko and Etzioni2008] Banko, Michele and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *ACL-08*.

[Banko et al.2007] Banko, Michele, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of IJCAI-07*.

[Cafarella et al.2007a] Cafarella, Michael, Dan Suciu, and Oren Etzioni. 2007a. Navigating extracted data with schema discovery. In *Proceedings of WWW-07*.

[Cafarella et al.2007b] Cafarella, Michael J., Christopher Re, Dan Suciu, Oren Etzioni, and Michele Banko. 2007b. Structured querying of web text: A technical challenge. In *Proceedings of CIDR-07*.

[Cohen and McCallum2003] Cohen, William and Andrew McCallum. 2003. Information extraction from the World Wide Web (tutorial). In *KDD*.

[Davidov and Rappoport2008] Davidov, Dmitry and Ari Rappoport. 2008. Unsupervised discovery of generic relationships using pattern clusters and its evaluation by automatically generated sat analogy questions. In *ACL-08*.

[Downey et al.2005] Downey, Doug, Oren Etzioni, and Stephen Soderland. 2005. A probabilistic model of redundancy in information extraction. In *Proceedings of IJCAI-05*.

[Erkan and Radev2004] Erkan, Güneş and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *JAIR*, 22:457–479.

[Etzioni et al.2004] Etzioni, Oren, Michael J. Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2004. Web-scale information extraction in KnowItAll. In *Proceedings of WWW-04*.

[Etzioni et al.2005] Etzioni, Oren, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, 165:91–134.

[Hassan et al.2007] Hassan, Samer, Rada Mihalcea, and Carmen Banea. 2007. Random-walk term weighting for improved text classification. *ICSC*.

[Hearst1992] Hearst, Marti A. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of COLING-92*.

[Kleinberg1999] Kleinberg, Jon Michael. 1999. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632.

[Lenat1995] Lenat, Douglas B. 1995. Cyc: a large-scale investment in knowledge infrastructure. *Commun. ACM*, 38(11).

[Liu and Yang2008] Liu, Nathan and Qiang Yang. 2008. Eigenrank: a ranking-oriented approach to collaborative filtering. In *SIGIR 2008*.

[Matuszek et al.2005] Matuszek, Cynthia, Michael Witbrock, Robert C. Kahlert, John Cabral, Dave Schneider, Purvesh Shah, and Doug Lenat. 2005. Searching for common sense: Populating cyc from the web. In *AAAI-05*.

[Mintz et al.2009] Mintz, Mike, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL-09*.

[Paşca et al.2006] Paşca, Marius, Dekang Lin, Jeffrey Bigham, Andrei Lifchits, and Alpa Jain. 2006. Organizing and searching the world wide web of facts - step one: The one-million fact extraction challenge. In *Proceedings of AAAI-06*.

[Page et al.1999] Page, Lawrence, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: Bringing order to the Web. Technical Report 1999/66, Stanford University, Computer Science Department.

[Pantel and Pennacchiotti2006] Pantel, Patrick and Marco Pennacchiotti. 2006. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *ACL/COLING-06*.

[Pantel et al.2004] Pantel, Patrick, Deepak Ravichandran, and Eduard Hovy. 2004. Towards terascale knowledge acquisition. In *COLING-04*.

[Pantel et al.2009] Pantel, Patrick, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *EMNLP-09*.

[Ravichandran and Hovy2002] Ravichandran, Deepak and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of ACL-08*, pages 41–47. Association for Computational Linguistics.

[Riloff and Jones1999] Riloff, Ellen and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of AAAI-99*.

[Talukdar et al.2008] Talukdar, Partha Pratim, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. Weakly-supervised acquisition of labeled class instances using graph random walks. In *Proceedings of EMNLP-08*.

[Yan et al.2009] Yan, Yulan, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. 2009. Unsupervised relation extraction by mining wikipedia texts with support from web corpus. In *ACL-09*.

Open Entity Extraction from Web Search Query Logs

Alpa Jain
Yahoo! Labs
alpa@yahoo-inc.com

Marco Pennacchiotti
Yahoo! Labs
pennac@yahoo-inc.com

Abstract

In this paper we propose a completely unsupervised method for open-domain entity extraction and clustering over query logs. The underlying hypothesis is that classes defined by mining search user activity may significantly differ from those typically considered over web documents, in that they better model the *user space*, i.e. users' perception and interests. We show that our method outperforms state of the art (semi-)supervised systems based either on web documents or on query logs (16% gain on the clustering task). We also report evidence that our method successfully supports a real world application, namely keyword generation for sponsored search.

1 Introduction

Search engines are increasingly moving beyond the traditional keyword-in document-out paradigm, and are improving user experience by focusing on user-oriented tasks such as query suggestions and search personalization. A fundamental building block of these applications is recognizing structured information, such as, *entities* (e.g., mentions of people, organizations, or locations) or relations among entities (Cao et al., 2008; Hu et al., 2009). For this, search engines typically rely on large collections of entities and relations built using information extraction (IE) techniques (Chaudhuri et al., 2009).

Commonly used IE techniques follow two main assumptions: (1) IE focuses on extracting information from syntactically and semantically "well-formed" pieces of texts, such as, news corpora and web documents (Pennacchiotti and Pantel, 2009); (2) extraction processes are bootstrapped with some pre-existing knowledge of the target domain (e.g entities are typically extracted for pre-defined categories, such as Actors, Manufacturers, Persons, Locations (Grishman and Sundheim, 1996)). Prior work (Banko et al., 2007), has looked into relaxing the second assumption and proposed open information extraction (OIE), a domain-independent and scalable extraction paradigm, which however focuses mostly on web corpora.

In this paper, we argue that for user-oriented applications discussed earlier, IE techniques should go beyond the traditional approach of using "well-formed" text documents. With this in mind, we explore the utility of search query logs, a rich source of user behaviors and perception, and build techniques for open entity extraction and clustering over query logs. We hypothesize that web documents and query logs model two different spaces: web documents model the *web space*, i.e. general knowledge about entities and concepts in an objective and generic way; search query logs model the *user space*, i.e. the users' view and perception of the world in a more specific fashion, where available information directly expresses users' needs and intents. For example, in a web space, 'britney spears' will tend to be similar and be clustered with other singers, such as 'celine dion' and 'bruce springsteen'. On the contrary, in the users' space, she is highly similar and clustered with other gossiped celebrities like 'paris hilton' and 'serena williams': the users' space better models the users' perception of that person; such a space is then highly valuable for all those applications where users' perceptions matters.

To computationally model our hypothesis for OIE over search query logs, we present a two phase approach to OIE for search query logs. The first phase (*entity extraction*) extracts entities from the search query logs using an unsupervised approach, by applying pattern-based heuristics and statistical measures. The second phase (*entity clustering*) induces classes over these entities by applying clustering techniques. In summary, our main contribu-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 510–518,
Beijing, August 2010

tions are: (1) We propose and instantiate a novel model for open information extraction over web search query logs; and we apply it to the task of entity extraction and clustering. (2) We show how we characterize each extracted entity to capture the 'user space', and induce classes over the entities. (3) We present an extensive evaluation over real-life datasets showing that query logs is a rich source for domain-independent user-oriented extraction tasks (Section 3). We also show the practicality of our approach by incorporating it into a real-world application, namely keyword suggestions for sponsored search (Section 4).

2 Open Entity Extraction on Query Log

In this section, we present our method for open entity extraction from query logs. We first describe our heuristic method for extracting entities (Section 2.1), and then three different feature 'user spaces' to cluster the entities (Section 2.2).

2.1 Entity Extraction

In our setting, entities correspond to Named Entities. i.e. they are defined using the standard named entity types described in (Sekine et al., 2002)[1]. In this paper, we use a set of entities extracted from query log, obtained by applying a simple algorithm (any other query log entity extraction method would apply here, e.g. (Pasca, 2007b)). The algorithm is based on the observation that oftentimes users construct their search query by copy-pasting phrases from existing texts. Due to this phenomenon, user queries often carry over surface-level properties such as capitalization and tokenization information. Our approach realizes this observation by identifying contiguous capitalized words from a user query. (In our experiments, we observed that 42% of the queries had at least one upper-case character.) Specifically, given a query $Q = q_1\ q_2\ q_3 \cdots\ q_n$, we define a candidate entity $E = e_1\ e_2\ \cdots\ e_m$ as the maximal sequence of words (i.e., alpha-numeric characters) in the query such that each word e_i in the entity begins with an uppercase character. The set of candidate entities is then cleaned by applying a set of heuristics, thus producing the final set of entities. In particular, for each extracted entity, we assign two confidence scores: a Web-based *representation score* and a query-log-based *standalone score*. The representation score checks if the case-sensitive representation observed for E in Q, is the most likely representation for E, as observed on a Web corpus (e.g., 'DOor HANGing TIps' is assigned a low representation score). The standalone score is based on the observation that a candidate E should often occur in a standalone form among the search query logs, in order to get the status of a proper named entity as defined in (Sekine et al., 2002; Grishman and Sundheim, 1996). In practice, among the query logs we must find queries of the form $Q == E$, capturing the fact that users are looking to learn more about the given entity[2].

2.2 Entity Clustering

The clustering phase takes as input any of the feature spaces presented in the rest of this section, and groups the entities according to the similarity of their vectors in the space. The desiderata for a clustering algorithm for the task of open-domain information extraction are the following: (1) The algorithm must be highly scalable, efficient, and able to handle high dimensionality, since the number of queries and the size of the feature vectors can be large; (2) We do not know in advance the number of clusters; therefore, the algorithm needs not to require a pre-defined number of clusters.

Any clustering algorithm fulfilling the above requirements would fit here. In our experiments, we adopt a highly scalable Map-Reduce implementation of the hard-clustering version of Clustering by Committee (CBC), a state-of-the-art clustering algorithm presented in (Pantel and Lin, 2002).

Context Feature Space. The basic hypothesis for the *context feature space*, is that an entity can be effectively represented by the set of contexts in which it appears in queries. This allows to capture the users' view of the entity, i.e. what people query, and want to know about the entity. This is similar to that proposed by Pasca (2007b; 2007a), i.e. that queries provide good semantics cues for modeling named entities.

Our query log feature space may significantly differ from a classical contextual feature space com-

[1]We exclude 'Time' and 'Numerical Expressions', which are out of the scope of our study.

[2]We refer the readers to (Jain and Pennacchiotti, 2010) for details on the entity extraction algorithms.

puted over a Web corpus, since the same entity can be differently perceived and described in the two corpora (query log and Web). Consider for example the entity 'galapagos islands'. Typical contexts on the Web and query log for this entity are:

web:	`endemic birds`
web:	`big turtles`
web:	`charles darwin foundation`
web:	`sensitive water`
qlog:	`trip to`
qlog:	`diving`
qlog:	`where are the`
qlog:	`travel package`

The difference between the two representations implies that entities that are similar on the Web, are not necessarily similar on query logs. For example, on the Web 'galapagos islands' is very similar to other countries such as 'tasmania', 'guinea' and 'luxemburg'; while on query log is similar to other sea-side travel destination and related concepts, such as 'greek isle', 'kauai snorkeling' and 'south america cruise'. Our new similarity computed over query log, is potentially useful for those applications in which is more important to represent users' intents, than an objective description of entities (e.g. in query suggestion and intent modeling).

To obtain our contextual representation we proceed as follows. For each entity e, we identify all queries in the query log, in which e appears. Then, we collect the set of all suffixes and postfixes of the entity in those queries. For example, given the entity 'galapagos islands' and the query 'summer 2008 galapagos islands tour', the contexts are: 'summer 2008' and 'tour'.

Once the set of all contexts of all entities has been collected, we discard contexts appearing less than τ-times in the query log, so to avoid statistical biases due to data sparseness (in the reported experiments we set $\tau = 200$). We then compute the corrected pointwise mutual information (*cpmi*) (Pantel and Ravichandran, 2004) between each instance and each context c as:

$$cpmi(e,c) = log_2 \frac{f(e,c) \cdot f(*,*)}{f(e) \cdot f(c)} \cdot M \qquad (1)$$

where $f(e,c)$ is the number of times e and c occur in the same query; $f(e)$ and $f(c)$ is the count of the entity and the context in the query log; $f(*,*)$ the overall count of all co-occurrences between contexts and entities; and M is the correction factor presented in (Pantel and Ravichandran, 2004), that eases the pmi's bias towards infrequent entities/features. Each instance is then represented in the feature space of all contexts, by the computed pmi values. Note that our method does not use any NLP parsing, since queries rarely present syntactic structure. This guarantees the method to be computationally inexpensive and easily adaptable to languages other than English.

Clickthrough Feature Space. During a search session, users issue a search query for which the search engine presents a list of result urls. Of the search results, users choose those urls that are representative of their intent. This interaction is captured by means of a click, which is logged by most search engines as *click-through data*. For instance, a search log may contain the following clicked urls for a query 'flv converter', for different users:

$user_1$:	`www.flv-converter.com`
$user_2$:	`www.videoconverterdownload.com/flv/`
$user_3$:	`www.ripzor.com/flv.html`

Our main motivation behind clustering entities based on past user click behavior is that non-identical queries that generate clicks on the same urls capture similar user intent. Thus, grouping entities that were issued as a query and generated user clicks on the same url may be considered similar. For instance, the query 'convert flv' may also generate clicks on one of the above urls, thus hinting that the two entities are similar. We observed that websites tend to dedicate a url per entity. Therefore, grouping by click urls can lead to clusters with synonyms (i.e., different ways of representing the same entity) or variants (e.g., spelling errors). To get more relevant clusters, instead of grouping entities by the click urls, we use the base urls. For instance, the url `www.ripzor.com/flv.html` is generalized to `www.ripzor.com`.

With the advent of encyclopedic websites such as, `www.wikipedia.org` and `wwww.youtube.com`, naively clustering entities by the clickthrough data can led to non-similar entities to be placed in the same cluster. For instance, we observed the most frequently clicked base url for both 'gold retriever' and 'abraham lincoln' is `www.wikipedia.org`. To address this issue, in our experiments we employed a stop-list by eliminating top-5 urls based on their inverse document frequency, where an entity is intended as the 'document'.

In practice, each extracted entity e is represented by a feature vector of size equal to the number of distinct base urls in the click-through data, across all users. Each dimension in the vector represents a url in the click-through information. The value f of an entity e for the dimension associated with url j is computed as:

$$f(e,j) = \begin{cases} \frac{w(e,j)}{\sqrt{\sum_i^{|\mathcal{U}|} w(e,i)^2}} & \text{if url } j \text{ clicked for query } e; \\ 0 & \text{otherwise.} \end{cases}$$

where $\mathcal{U}$ is the set of base urls found in click-through data when entity e was issued as a query; and $w(e,i)$ is the number of time the base url i was clicked when e was a query.

Hybrid Feature Space. We also experiment a hybrid feature space, which is composed by the normalized union of the two feature spaces above (i.e. context and clickthrough). Though more complex hybrid models could be applied, such as one based on ensemble clustering, we here opt for a simple solution which allows to better read and compare to other methods.

3 Experimental Evaluation

In this section, we report experiments on our clustering method. The goal of the experiment is two-fold: (1) evaluate the intrinsic quality of the clustering methods, i.e. if two entities in the same cluster are similar or related from a web user's perspective; (2) verify if our initial hypothesis holds, i.e. if query log based features spaces capture different properties than Web based feature spaces (i.e. the 'user space'). In Section 3.1 we describe our experimental setup; and, in 3.2 we provide the results. We couple this intrinsic evaluation with an extrinsic application-driven one in Section 4.

3.1 Experimental Settings

In the experiments we use the following datasets:

Query log: A random sample of 100 million, fully anonymized queries collected by the Yahoo! search engine in the first 3 months of 2009, along with their frequency. This dataset is used to generate both the context and the clickthrough feature spaces for the clustering step.

Web documents: A collection of 500 million web pages crawled by a Yahoo! search engine crawl. This data set is used to implement a web-based feature space that we will compare to in Section 3.2.

Entity set: A collection of 2,067,385 entities, extracted with the method described in 2.1, which shows a precision of 0.705 $\pm$0.044. Details on the evaluation of such method are available in (Jain and Pennacchiotti, 2010), where a full comparison with state-of-the-art systems such as (Pasca, 2007b) and (Banko et al., 2007) are also reported.

Evaluation methodology: Many clustering evaluation metrics have been proposed, ranging from Purity to Rand-statistics and F-Measure. We first select from the original 2M entity set, a random set of n entities biased by their frequency in query logs, so to keep the experiment more realistic (more frequent entities have more chances to be picked in the sample). For each entity e in the sample set, we derived a random list of k entities that are clustered with e. In our experiments, we set $n = 10$ and $k = 20$. We then present to a pool of paid editors, each entity e along with the list of co-clustered entities. Editors are requested to classify each co-clustered entity e_i as *correct* or *incorrect*. An entity e_i is deemed as correct, if it is similar or related to e from a web user's perspective: to capture this intuition, the editor is asked the question: 'If you were interested in e, would you be also interested in e_i in any intent?'.[3] Annotators' agreement over a random set of 30 entities is $kappa = 0.64$ (Marques De Sá, 2003), corresponding to substantial agreement. Additionally, we ask editors to indicate the *relation type* between e and e_i (synonyms, siblings, parent-child, topically related).

Compared methods:

`CL-CTX`: A CBC run, based on the query log context feature space (Section 2.2).

`CL-CLK`: A CBC run, based on the clickthrough feature space (Section 2.2).

[3]For example, if someone is interested in 'hasbro', he could be probably also be interested in 'lego', when the intent is buying a toy. The complete set of annotation guidelines is reported in (Jain and Pennacchiotti, 2010).

method	# cluster	avg cluster size
CL-Web	1,601	240
CL-CTX	875	1,182
CL-CLK	4,385	173
CL-HYB	1,580	478

Table 1: Statistics on the clustering results.

CL-HYB: A CBC run, based on the hybrid space that combines CL-CTXand CL-CLK(Section 2.2).
CL-Web: A state-of-the-art open domain method based on features extracted from the Web documents data set (Pantel et al., 2009). This method runs CBC over a space where features are the contexts in which an entity appears (noun chunks preceding and following the target entity); and feature value is the *pmi* between the entity and the chunks.

Evaluation metrics: We evaluate each method using **accuracy**, intended as the percentage of correct judgments.

3.2 Experimental Results

Table 3 reports accuracy results. CL-HYB is the best performing method, achieving 0.85 accuracy, respectively +4% and +11% above CL-CLK and CL-Web. CL-CTX shows the lowest performance. Our results suggest that query log spaces are more suitable to model the 'user space' wrt web features. Specifically, clickthrough information are most useful confirming our hypothesis that queries that generate clicks on the same urls capture similar user intents.

To have an anecdotal and practical intuition on the results, in Table 2 we report some entities and examples of other entities from the same clusters, as obtained from the CL-HYB and CL-Web methods. The examples show that CL-HYB builds clusters according to a variety of relations, while CL-Web mostly capture sibling-like relations.

One relevant of such relations is topicality. For example, for 'aaa insurance' the CL-HYB cluster mostly contains entities that are topically related to the American Automobile Association, while the CL-Web cluster contains generic business companies. In this case, the CL-HYB approach simply chose to group together entities having clicks to 'aaa.com' and appearing in contexts as 'auto club'. On the contrary, CL-Web grouped according to contexts such as 'selling' and 'company'. The entity 'hip osteoarthritis' shows a similar behavior: CL-HYB groups entities topically related to orthopedic issues, since most of the entities are sharing contexts such as 'treatment' and 'recovery' and, at the same time, clicks to urls such as 'orthoinfo.aaos.org' and 'arthirtis.about.com'.

entity	CL-HYB	CL-Web
aaa insurance	roadside assistance personal liability insurance international driving permits aaa minnesota travelers checks	loanmax pilot car service localnet fibermark country companies insurance
paris hilton	brenda costa adriana sklenarikova kelly clarkson anja rubik federica ridolfi	julia roberts brad pitt nicole kidman al pacino tom hanks
goldie hawn	bonnie hunt brad pitt tony curtis nicole kidman nicholas cage	julia roberts brad pitt nicole kidman al pacino tom hanks
basic algebra	numerical analysis discrete math lattice theory nonlinear physics ramsey theory	math tables trigonometry help mathtutor surface area formula multiplying fractions
hip osteoarthritis	atherosclerosis pneumonia hip fracture breast cancer anorexia nervosa	wrist arthritis disc replacement rotator cuff tears shoulder replacement american orthopedic society
acer america	acer aspire accessories aspireone acer monitors acer customer service acer usa	microsoft casio computer borland software sony nortel networks

Table 2: Sample of the generated entity clusters.

Another interesting observation regards entities referring to people. The 'paris hilton' and 'goldie hawn' examples show that the CL-Web approach groups famous people according to their category – i.e. profession in most cases. On the contrary, query log approaches tend to group people according to their social attitude, when this prevails over the profession. In the example, CL-HYB clusters the actress 'goldie hawn' with other actors, while 'paris hilton' is grouped with an heterogeneous set of celebrities that web users tend to query and click in a same manner: In this case, the social persona of 'paris hilton' prevails over its profession (actress/singer). This aspect is important in many applications, e.g. in query suggestion, where one wants to propose to the user entities that have been similarly queried and clicked.

In order to check if the above observations are not anecdotal, we studied the *relation type* annotation provided by the editors (Table 4). Table shows

method	Precision
CL-Web	0.735
CL-CTX	0.460
CL-CLK	0.815 †
CL-HYB	**0.850** †

Table 3: Precision of various clustering methods († indicates statistical-significant better than the CL-Web method, using t-test).

that query log based methods are more varied in the type of clusters they build. Table 5 shows the difference between the clustering obtained using the different methods and the overlap between the produced clusters. For example, 40% of the relations for the CL-HYB system are topical, while 32% are sibiling ones. On the contrary, the CL-Web method is highly biased towards sibling relations.

As regard a more attentive analysis of the different query log based methods, CL-CTX has the lowest performance. This is mainly due to the fact that contextual data are sometimes too sparse and generic. For example 'mozilla firefox' is clustered with 'movie program' and 'astro reading' because they share only some very generic contexts such as 'free downloads'. In order to get more data, one option is to relax the τ threshold (see Section 2) so to include more contexts in the semantic space. Unfortunately, this would have a strong drawback, in that low-frequency context tend to be idiosyncratic and spurious. A typical case regards recurring queries submitted by robots for research purposes, such as 'who is X', 'biography of X', or 'how to X'. These queries tend to build too generic clusters containing people or objects. Another relevant problem of the CL-CTX method is that even when using a high τ cut, clusters still tend to be too big and generic, as statistics in Table 4 shows.

CL-CTX, despite the low performance, is very useful when combined with CL-CLK. Indeed the CL-HYB system improves +4% over the CL-CLK system alone. This is because the CL-HYB method is able to recover some misleading or incomplete evidence coming from the CL-CLK using features provided by CL-CLK. For example, editors judged as incorrect 11 out of 20 entities co-clustered with the entity 'goldie hawn' by CL-CLK. Most of these errors are movies (e.g. 'beverly hills cops') soap operas (e.g. 'sortilegio') and directors, because all have clicks to 'imdb.com' and 'movies.yahoo.com'.

class	method			
	CL-Web	CL-CTX	CL-CLK	CL-HYB
topic	0.27	0.46	0.46	0.40
sibling	0.72	0.43	0.29	0.32
parent	-	0.09	0.13	0.09
child	0.01	-	0.01	0.02
synonym	0.01	0.03	0.12	0.16

Table 4: Fraction of entities that have been classified by editors in the different relation types.

method	labelled clusters			
	CL-CTX	CL-CLK	CL-HYB	CL-Web
CL-CTX	-	0.2	0.53	0.29
CL-CLK	0.21	-	0.54	0.34
CL-HYB	0.53	0.51	-	0.31
CL-Web	0.33	0.35	0.41	-

Table 5: Purity of clusters for each method using clusters from other methods as "labelled" data.

CL-HYB recovers these errors by including features coming from CL-CTX such as 'actress'.

In summary, query log spaces group together entities that are similar by web users (this being topical similarity or social attitude), thus constituting a practical model of the 'user space' to be leveraged by web applications.

4 Keywords for Sponsored Search

In this section we explore the use of our methods for keyword generation for sponsored search. In sponsored search, a search company opens an auction, where on-line advertisers bid on specific keywords (called *bidterms*). The winner is allowed to put its ad and link on the search result page of the search company, when the bidterm is queried. Companies such as Google and Yahoo are investing efforts for improving their bidding platforms, so to attract more advertisers in the auctions. Bidterm suggestion tools (adWords, 2009; yahooTool, 2009) are used to help advertiser in selecting bidterms: the advertisers enters a seed keyword (*seed*) expressing the intent of its ad, and the tool returns a list of suggested keywords (*suggestions*) that it can use for bidding – e.g for the seed 'mp3 player', a suggestion could be 'ipod nano'. The task of generating bid suggestions (i.e. *keyword generation*) is typically automatic, and has received a growing attention in the search community for its impact on search company revenue. The main problem of existing methods for suggestion (adWords, 2009; yahooTool, 2009; wordTracker, 2009) is that they produce only suggestions that contain the initial seed (e.g. 'belkin mp3 player' for the seed 'mp3 player'), while nonobvious (and potentially less expensive) suggestions not containing the seed are neglected (e.g. 'ipod nano' for 'mp3 player'). For example for 'galapagos islands', a typical production system suggests 'galapagos islands tour' which cost almost 5$ per click; while the less obvious 'isla santa cruz' would cost only 0.35$. Below we show our method to discover such nonobvious suggestions, by retrieving entities in the same cluster of a given seed.

4.1 Experimental Setting

We evaluate the quality of the suggestions proposed by different methods for a set of seed bidterms., adopting the evaluation schema in (Joshi and Motwani, 2006)

Dataset Creation. To create the set of seeds, we use Google skTool[4]. The tool provides a list of popular bid terms, organized in a taxonomy of advertisement topics. We select 3 common topics: tourism, vehicles and consumer-electronics. For each topic, we randomly pick 5 seeds among the 800 most popular bid terms, which also appear in our entity set described in Section 3.1.[5]. We evaluate a system by collecting all its suggestions for the 15 seeds, and then extracting a random sample of 20 suggestions per seed.

Evaluation and Metrics. We use precision and Nonobviousness. **Precision** is computed by asking two experienced human experts to classify each suggestion of a given seed, as *relevant* or *irrelevant*. A suggestion is deemed as relevant if any advertiser would likely choose to bid for the suggestion, having as intent the seed. Annotator agreement, evaluated on a subset of 120 suggestions is $kappa = 0.72$ (substantial agreement). Precision is computed as the percentage of suggestions judged as relevant. **Nonobviousness** is a metric introduced in (Joshi and Motwani, 2006), capturing how nonobvious the suggestions are. It simply counts how many suggestions for a given seed do not contain the seed itself (or any of its variants): this metric is computed automatically using string matching and a simple stemmer.

Comparisons. We compare the suggestions proposed by `CL-CTX`, `CL-CLK`, and `CL-HYB`, against `Web` and two reference state-of-the-art production systems: Google AdWords (`GOO`) and Yahoo Search Marketing Tool (`YAH`). As concerns our methods, we extract as suggestions the entities that occur in the same cluster of a given seed. For the production systems, we rely on the suggestions proposed on the website of the tools.

4.2 Experimental Results

Precision results are reported in the second column of Table 6. Both `CL-CLK` and `CL-HYB` outperform `Web` in precision, `CL-HYB` being close to the upper-bound of the two production systems. As expected, production systems show a very high precision but their suggestions are very obvious. Our results are fairly in line with those obtained on a similar dataset, by Joshi and Motwani (2006).

A closer look at the results shows that most of the errors for `CL-CTX` are caused by the same problem outlined in Section 3.2: Some entities are wrongly assigned to a cluster, because they have some high $cpmi$ context feature which is shared with the cluster centroid, but which is not very characteristic for the entity itself. This is particularly evident for some of the low frequency entities, where $cpmi$ values could not reflect the actual semantics of the entity. For example the entity 'nickelodeon' (a kids tv channel in UK) is assigned to the cluster of 'galapagos islands', because of the feature 'cruise': indeed, some people query about 'nickelodeon cruise' because the tv channel organizes some kids cruises. Other mistakes are due to feature ambiguity. For example, the entity 'centurion boats' is assigned to the cluster of 'obertauern' (a ski resort in Austria), because they share the ambiguous feature 'ski' (meaning either winter-ski or water-ski). As for the `CL-CLK` system, some of the errors are caused by the fact that some base url can refer to very different types of entities. For example the entity 'color copier' is suggested for the the camera 'canon rebel xti', since they both share clicks to the Canon website. The `CL-HYB` system achieves a higher preci-

[4] http://www.google.com/sktool

[5] The final set of 15 bid terms is: *tourism:*galapagos islands,holiday insurance,hotel booking,obertauern,wagrain; *vehicles:*audi q7,bmw z4,bmw dealers,suzuki grand vitara,yamaha banshee; *consumer electr:*canon rebel xti,divx converter,gtalk,pdf reader,flv converter.

method	Precision	Nonobviousness
GOO	0.982	0.174
YAH	0.966	0.195
Web	0.814	0.827
CL-CTX	0.547	0.963
CL-CLK	0.827	0.630
CL-HYB	0.946	0.567

Table 6: Results for keyword generation.

sion wrt CL-CTX and CL-CLK: the combination of the two spaces decreases the impact of misleading features –e.g. for 'yamaha bunshee', all CL-HYB 's suggestions are correct, while almost all CL-CLK 's suggestions are incorrect: the hybrid system recovered the negative effect of the misleading feature ebay.com, by backing up on features from the contextual subspace (e.g. 'custom', 'specs', 'used parts').

Nonobviousness results are reported in column three of Table 6. All our systems return a high number of nonobvious suggestions (all above 50%).[6] On the contrary, GOO and YAH show low performance, as both systems are heavily based on the substring matching technique. This strongly motivates the use of semantic approaches as those we propose, that guarantee at the same time both a higher linguistic variety and an equally high precision wrt the production systems. For example, for the seeds 'galapagos islands', GOO returns simple suggestions such as 'galapagos islands vacations' and 'galapagos islands map'; while CL-HYB returns 'caribbean mexico' and 'pacific dawn', two terms that are semantically related but dissimilar from the seed. Remember that these letter terms are related to the seed because they are similar in the user space, i.e. users looking at 'galapagos islands' tend to similarly look for 'caribbean mexico' and 'pacific dawn'. These suggestions would then be very valuable for tourism advertisers willing to improve their visibility through a non-trivial and possibly less expensive set of bid terms.

5 Related Work

While literature abounds with works on entity extraction from web documents (e.g. (Banko et al., 2007; Chaudhuri et al., 2009; Pennacchiotti and Pantel, 2009)), the extraction of classes of entities over query logs is a pretty new task, recently introduced in (Pasca, 2007b). Pasca's system extracts entities of pre-defined classes in a semi-supervised fashion, starting with an input class represented by a set of seeds, which are used to induce typical query-contexts for the class. Contexts are then used to extract and select new candidate instances for the class. A similar approach is also adopted in (Sekine and Suzuki, 2007). Pasca shows an improvement of about 20% accuracy, compared to existing Web-based systems. Our extraction algorithm differs from Pasca's work in that it is completely unsupervised. Also, Pasca's cannot be applied to OIE, i.e. it only works for pre-defined classes. Our clustering approach is related to Lin and Wu's work (Lin and Wu, 2009). Authors propose a semi-supervised algorithm for query classification. First, they extract a large set of 20M phrases from a query log, as those unique queries appearing more than 100 times in a Web corpus. Then, they cluster the phrases using the K-means algorithm, where features are the phrases' bag-of-words contexts computed over a web corpus. Finally, they classify queries using a logistic regression algorithm. Our work differs from Lin and Wu, as we focus on entities instead of phrases. Also, the features we use for clustering are from query logs and click data, not web contexts.

6 Conclusions

We presented an open entity extraction approach over query logs that goes beyond the traditional web corpus, with the goal of modeling a 'user-space' as opposed to an established 'web-space'. We showed that the clusters generated by query logs substantially differ from those by a Web corpus; and that our method is able to induce state-of-the-art quality classes on a user-oriented evaluation on the real world task of keyword generation for sponsored search. As future work we plan to: (i) experiment different clustering algorithms and feature models, e.g. soft-clustering for handling ambiguous entities; (ii) integrate the Web space and the query log spaces; (iii) embed our methods in in existing tools for intent modeling, query suggestion and similia, to check its impact in production systems.

[6] Note that very high values for CL-CTX may be misleading, as many of the suggestions proposed by this system are incorrect (see precision results) and hence non-obvious (e.g., 'derek lewis' for 'galapagos islands').

References

adWords. 2009. Google adwords. adwords.google.com/select/keywordtoolexternal.

Banko, Michele, Michael Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of IJCAI*.

Cao, Huanhuan, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of KDD-08*.

Chaudhuri, Surajit, Venkatesh Ganti, and Dong Xin. 2009. Exploiting web search to generate synonyms for entities. In *Proceedings of WWW-09*.

Grishman, R. and B. Sundheim. 1996. Message understanding conference- 6: A brief history. In *Proceedings of COLING*.

Hu, Jian, Gang Wang, Fred Lochovsky, Jian tao Sun, and Zheng Chen. 2009. Understanding user's query intent with Wikipedia. In *Proceedings of WWW-09*.

Jain, Alpa and Marco Pennacchiotti. 2010. Open Information Extraction from Web Search Query Logs. Technical Report YL-2010-003, Yahoo! Labs.

Joshi, Amruta and Rajeev Motwani. 2006. Keyword generation for search engine advertising. In *Proceedings of Sixth IEEE-ICDM*.

Lin, Dekang and Xiaoyun Wu. 2009. Phrase clustering for discriminative learning. In *Proceedings of ACL-IJCNLP-2009*.

Marques De Sá, Joaquim P. 2003. *Applied Statistics*. Springer Verlag.

Pantel, Patrick and Dekang Lin. 2002. Discovering word senses from text. In *Proceedings KDD-02*.

Pantel, Patrick and Deepak Ravichandran. 2004. Automatically labeling semantic classes. In *Proceeding of HLT-NAACL-2004*.

Pantel, Patrick, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, and Vishnu Vyas. 2009. Web-scale distributional similarity and entity set expansion. In *Proceedings of EMNLP-09*.

Pasca, Marius. 2007a. Organizing and searching the world wide web of facts - step two: Harnessing the wisdom of the crowds. In *Proceedings of the WWW-2007*.

Pasca, Marius. 2007b. Weakly-supervised discovery of named entities using web search queries. In *Proceedings of CIKM-2007*.

Pennacchiotti, Marco and Patrick Pantel. 2009. Entity extraction via ensemble semantics. In *Proceedings of EMNLP-2009*.

Sekine, Satoshi and Hisami Suzuki. 2007. Acquiring ontological knowledge from query logs. In *Proceedings of WWW-07*.

Sekine, Satoshi, Kiyoshi Sudo, and Chikashi Nobata. 2002. Extended named entity hierarchy. In *Proceedings of LREC-2002*.

wordTracker. 2009. Word tracker. www.wordtracker.com.

yahooTool. 2009. Yahoo search marketing. searchmarketing.yahoo.com.

Reranking Models in Fine-grained Opinion Analysis

Richard Johansson and **Alessandro Moschitti**
University of Trento
{johansson, moschitti}@disi.unitn.it

Abstract

We describe the implementation of reranking models for fine-grained opinion analysis – marking up opinion expressions and extracting opinion holders. The reranking approach makes it possible to model complex relations between multiple opinions in a sentence, allowing us to represent how opinions *interact* through the syntactic and semantic structure. We carried out evaluations on the MPQA corpus, and the experiments showed significant improvements over a conventional system that only uses local information: for both tasks, our system saw recall boosts of over 10 points.

1 Introduction

Recent years have seen a surge of interest in the automatic processing of *subjective language*. The technologies emerging from this research have obvious practical uses, either as stand-alone applications or supporting other NLP tools such as information retrieval or question answering systems. While early efforts in subjectivity analysis focused on coarse-grained tasks such as retrieving the subjective documents from a collection, most recent work on this topic has focused on fine-grained tasks such as determining the attitude of a particular person on a particular topic. The development and evaluation of such systems has been made possible by the release of manually annotated resources using fairly fine-grained representations to describe the structure of subjectivity in language, for instance the MPQA corpus (Wiebe et al., 2005).

A central task in the automatic analysis of subjective language is the indentification of *subjective expressions*: the text pieces that allow us to draw the conclusion that someone has a particular feeling about something. This is necessary for further analysis, such as the determination of opinion holder and the polarity of the opinion. The MPQA corpus defines two types of subjective expressions: *direct subjective expressions* (DSEs), which are explicit mentions of attitude, and *expressive subjective elements* (ESEs), which signal the attitude of the speaker by the choice of words. The prototypical example of a DSE would be a verb of statement or categorization such as *praise* or *disgust*, and the opinion holder would typically be a direct semantic argument of this verb. ESEs, on the other hand, are less easy to categorize syntactically; prototypical examples would include value-expressing adjectives such as *beautiful* and strongly charged words like *appeasement*, while the relation between the expression and the opinion holder is typically less clear-cut than for DSEs. In addition to DSEs and ESEs, the MPQA corpus also contains annotation for non-subjective statements, which are referred to as *objective speech events* (OSEs).

Examples (1) and (2) show two sentences from the MPQA corpus where DSEs and ESEs have been manually annotated.

(1) He [made such charges]$_{DSE}$ [despite the fact]$_{ESE}$ that women's political, social and cultural participation is [not less than that]$_{ESE}$ of men.

(2) [However]$_{ESE}$, it is becoming [rather fashionable]$_{ESE}$ to [exchange harsh words]$_{DSE}$ with each other [like kids]$_{ESE}$.

The task of marking up these expressions has usually been approached using straightforward sequence labeling techniques using using simple features in a small contextual window (Choi et al., 2006; Breck et al., 2007). However, due to

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 519–527,
Beijing, August 2010

the simplicity of the feature sets, this approach fails to take into account the fact that the semantic and pragmatic interpretation of sentences is not only determined by words but also by syntactic and shallow-semantic *relations*. Crucially, taking grammatical relations into account allows us to model how expressions *interact* in various ways that influence their interpretation as subjective or not. Consider, for instance, the word *said* in examples (3) and (4) below, where the interpretation as a DSE or an OSE is influenced by the subjective content of the enclosed statement.

(3) "We will identify the $[\text{culprits}]_{ESE}$ of these clashes and $[\text{punish}]_{ESE}$ them," he $[\text{said}]_{DSE}$.

(4) On Monday, 80 Libyan soldiers disembarked from an Antonov transport plane carrying military equipment, an African diplomat $[\text{said}]_{OSE}$.

In addition, the various opinions expressed in a sentence are very interdependent when it comes to the resolution of their *holders*, i.e. determining the entity that harbors the sentiment manifested textually in the opinion expression. Clearly, the structure of the sentence is influential also for this task: an ESE will be quite likely to be linked to the same opinion holder as a DSE directly above it in the syntactic tree.

In this paper, we demonstrate how syntactic and semantic structural information can be used to improve the detection of opinion expressions and the extraction of opinion holders. While this feature model makes it impossible to use the standard sequence labeling method, we show that with a simple strategy based on *reranking*, incorporating structural features results in a significant improvement. In an evaluation on the MPQA corpus, the best system we evaluated, a reranker using the Passive–Aggressive learning algorithm, achieved a 10-point absolute improvement in soft recall, and a 5-point improvement in F-measure, over the baseline sequence labeler. Similarly, the recall is boosted by almost 11 points for the holder extraction (3 points in F-measure) by modeling the interaction of opinion expressions with respect to holders.

2 Related Work

Since the most significant body of work in subjectivity analysis has been dedicated to coarse-grained tasks such as document polarity classification, most approaches to analysing the sentiment of natural-language text have relied fundamentally on purely lexical information (see (Pang et al., 2002; Yu and Hatzivassiloglou, 2003), *inter alia*) or low-level grammatical information such as part-of-speech tags and functional words (Wiebe et al., 1999). This is not unexpected since these problems have typically been formulated as text categorization problems, and it has long been agreed in the information retrieval community that very little can be gained by complex linguistic processing for tasks such as text categorization and search (Moschitti and Basili, 2004).

As the field moves towards increasingly sophisticated tasks requiring a detailed analysis of the text, the benefit of syntactic and semantic analysis becomes more clear. For the task of subjective expression detection, Choi et al. (2006) and Breck et al. (2007) used syntactic features in a sequence model. In addition, syntactic and shallow-semantic relations have repeatedly proven useful for subtasks of subjectivity analysis that are inherently *relational*, above all for determining the holder or topic of a given opinion. Choi et al. (2006) is notable for the use of a global model based on hand-crafted constraints and an integer linear programming optimization step to ensure a globally consistent set of opinions and holders.

Works using syntactic features to extract topics and holders of opinions are numerous (Bethard et al., 2005; Kobayashi et al., 2007; Joshi and Penstein-Rosé, 2009; Wu et al., 2009). Semantic role analysis has also proven useful: Kim and Hovy (2006) used a FrameNet-based semantic role labeler to determine holder and topic of opinions. Similarly, Choi et al. (2006) successfully used a PropBank-based semantic role labeler for opinion holder extraction. Ruppenhofer et al. (2008) argued that semantic role techniques are useful but not completely sufficient for holder and topic identification, and that other linguistic phenomena must be studied as well. One such linguistic pheonomenon is the *discourse* structure, which has recently attracted some attention in the subjectivity analysis community (Somasundaran et al., 2009).

3 Modeling Interaction over Syntactic and Semantic Structure

Previous systems for opinion expression markup have typically used simple feature sets which have allowed the use of efficient off-the-shelf sequence labeling methods based on Viterbi search (Choi et al., 2006; Breck et al., 2007). This is not possible in our case since we would like to extract structural, relational features that involve *pairs* of opinion expressions and may apply over an arbitrarily long distance in the sentence.

While it is possible that search algorithms for exact or approximate inference can be constructured for the $\arg\max$ problem in this model, we sidestepped this issue by using a *reranking* decomposition of the problem:

- Apply a standard Viterbi-based sequence labeler based on local context features but no structural interaction features. Generate a small candidate set of size k.
- Generate opinion holders for every proposed opinion expression.
- Apply a complex model using interaction features to pick the top candidate from the candidate set.

The advantages of a reranking approach compared to more complex approaches requiring advanced search techniques are mainly simplicity and efficiency: this approach is conceptually simple and fairly easy to implement provided that k-best output can be generated efficiently, and features can be arbitrarily complex – we don't have to think about how the features affect the algorithmic complexity of the inference step. A common objection to reranking is that the candidate set may not be diverse enough to allow for much improvement unless it is very large; the candidates may be trivial variations that are all very similar to the top-scoring candidate.

3.1 Syntactic and Semantic Structures

We used the syntactic–semantic parser by Johansson and Nugues (2008) to annnotate the sentences with dependency syntax (Mel'čuk, 1988) and shallow semantic structures in the PropBank (Palmer et al., 2005) and NomBank (Meyers et al., 2004) frameworks. Figure 1 shows an example of the annotation: The sentence *they called him a liar*, where *called* is a DSE and *liar* is an ESE, has been annotated with dependency syntax (above the text) and PropBank-based semantic role structure (below the text). The predicate *called*, which is an instance of the PropBank frame `call.01`, has three semantic arguments: the Agent (A0), the Theme (A1), and the Predicate (A2), which are realized on the surface-syntactic level as a subject, a direct object, and an object predicative complement, respectively.

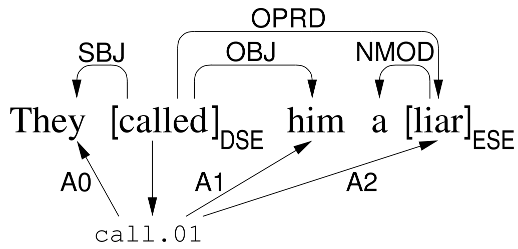


Figure 1: Syntactic and shallow semantic structure.

3.2 Base Sequence Labeling Model

To solve the first subtask, we implemented a standard sequence labeler for subjective expression markup, similar to the approach by Breck et al. (2007). We encoded the opinionated expression brackets using the IOB2 encoding scheme (Tjong Kim Sang and Veenstra, 1999) and trained the model using the metod by Collins (2002).

The sequence labeler used word, POS tag, and lemma features in a window of size 3. In addition, we used prior polarity and intensity features derived from the lexicon created by Wilson et al. (2005). It is important to note that prior subjectivity does not always imply subjectivity in a particular context; this is why contextual features are essential for this task.

This sequence labeler was used to generate the candidate set for the reranker. To generate reranking training data, we carried out a 5-fold hold-out procedure: We split the training set into 5 pieces, trained a sequence labeler on pieces 1 to 4, applied it to piece 5 and so on.

3.3 Base Opinion Holder Extractor

For every opinion expression, we extracted *opinion holders*, i.e. mentions of the entity holding the opinion denoted by the opinion expression. Since the problem of holder extraction is in many ways similar to semantic argument detection – when the opinion expression is a verb, finding the holder typically entails finding a SPEAKER argument – we approached this problem using methods inspired by semantic role labeling. We thus trained support vector machines using the LIBLINEAR software (Fan et al., 2008), and applied them to the noun phrases in the same sentence as the holder. Separate classifiers were trained to extract holders for DSEs, ESEs, and OSEs. The classifiers used the following feature set:

SYNTACTIC PATH. Similarly to the path feature widely used in SRL, we extract a feature representing the path in the dependency tree between the expression and the holder (Johansson and Nugues, 2008). For instance, the path from the DSE *called* to the holder *They* is `SBJ↓`.

SHALLOW-SEMANTIC RELATION. If there is a direct shallow-semantic relation between the expression and the holder, use a feature representing its semantic role, such as `A0` for *They* with respect to *called*.

EXPRESSION HEAD WORD AND POS.

HOLDER HEAD WORD AND POS.

DOMINATING EXPRESSION TYPE.

CONTEXT WORDS AND POS FOR HOLDER.

EXPRESSION VERB VOICE.

However, there are also differences compared to typical argument extraction in SRL. First, it is important to note that the MPQA corpus does not annotate direct links from opinions to a holders, but from opinions to *holder coreference chains*. To handle this issue, we created positive training instances for *all* members of the coreference chain in the same sentence as the opinion, and negative instances for the other noun phrases.

Secondly, an opinion may be linked not to an overt noun phrase in a sentence, but to an *implicit* holder; a special case of implicit holder is the *writer* of the text. We trained separate classifiers to detect these situations. These classifiers did not use the features requiring a holder phrase.

Finally, there is a restriction that every expression may have at most one holder, so at test time we select only the highest-scoring opinion holder candidate.

3.4 Opinion Expression Reranker Features

The rerankers use two types of structural features: syntactic features extracted from the dependency tree, and semantic features extracted from the predicate–argument (semantic role) graph.

The syntactic features are based on paths through the dependency tree. This creates a small complication for multiword opinion expressions; we select the shortest possible path in such cases. For instance, in example (1) above, the path will be computed between *made* and *despite*, and in (2) between *fashionable* and *exchange*.

We used the following syntactic interaction features:

SYNTACTIC PATH. Given a pair opinion expressions, we use a feature representing the labels of the two expressions and the path between them through the syntactic tree. For instance, for the DSE *called* and the ESE *liar* in Figure 1, we represent the syntactic configuration using the feature `DSE:OPRD↓:ESE`, meaning that the path from the DSE to the ESE follows an OPRD link downward.

LEXICALIZED PATH. Same as above, but with lexical information attached: `DSE/called:OPRD↓:ESE/liar`.

DOMINANCE. In addition to the features based on syntactic paths, we created a more generic feature template describing dominance relations between expressions. For instance, from the graph in Figure 1, we extract the feature `DSE/called→ESE/liar`, meaning that a DSE with the word *called* dominates an ESE with the word *liar*.

The semantic features were the following:

PREDICATE SENSE LABEL. For every predicate found inside an opinion expression, we add a feature consisting of the expression label and the predicate sense identifier. For instance, the verb *call* which is also a DSE is represented with the feature `DSE/call.01`.

PREDICATE AND ARGUMENT LABEL. For every argument of a predicate inside an opinion expression, we also create a feature representing the predicate–argument pair: `DSE/call.01:A0`.

CONNECTING ARGUMENT LABEL. When a predicate inside some opinion expression is connected to some argument inside another opinion expression, we use a feature consisting of the two expression labels and the argument label. For instance, the ESE *liar* is connected to the DSE *call* via an A2 label, and we represent this using a feature `DSE:A2:ESE`.

Apart from the syntactic and semantic features, we also used the score output from the base sequence labeler as a feature. We normalized the scores over the k candidates so that their exponentials summed to 1.

3.5 Opinion Holder Reranker Features

In addition, we modeled the interaction between different opinions with respect to their holders. We used the following two features to represent this interaction:

SHARED HOLDERS. A feature representing whether or not two opinion expressions have the same holder. For instance, if a DSE dominates an ESE and they have the same holder as in Figure 1 where the holder is *They*, we represent this by the feature `DSE:ESE:true`.

HOLDER TYPES + PATH. A feature representing the types of the holders, combined with the syntactic path between the expressions. The types take the following possible values: explicit, implicit, writer. In Figure 1, we would thus extract the feature `DSE/Expl:OPRD↓:ESE/Expl`.

Similar to base model feature for the expression detection, we also used a feature for the output score from the holder extraction classifier.

3.6 Training the Reranker

We trained the reranker using the method employed by many rerankers following Collins (2002), which learns a scoring function that is trained to maximize performance on the reranking task. While there are batch learning algorithms that work in this setting (Tsochantaridis et al., 2005), online learning methods have been more popular for performance reasons. We investigated two online learning algorithms: the popular *structured perceptron* (Collins, 2002) and the Passive–Aggressive (PA) algorithm (Crammer et al., 2006). To increase robustness, we used an averaged implementation (Freund and Schapire, 1999) of both algorithms.

The difference between the two algorithms is the way the weight vector is incremented in each step. In the perceptron, for a given input x, we update based on the difference between the correct output y and the predicted output $\hat{y}$, where Φ is the feature representation function:

$$\begin{aligned}&\hat{y} \leftarrow \arg\max_h w \cdot \Phi(x,h)\\&w \leftarrow w + \Phi(x,y) - \Phi(x,\hat{y})\end{aligned}$$

In the PA algorithm, which is based on the theory of large-margin learning, we instead find the $\hat{y}$ that violates the margin constraints maximally. The update step length τ is computed based on the margin; this update is bounded by a regularization constant C:

$$\begin{aligned}&\hat{y} \leftarrow \arg\max_h w \cdot \Phi(x,h) + \sqrt{\rho(y,h)}\\&\tau \leftarrow \min\left(C, \frac{w(\Phi(x,\hat{y})-\Phi(x,y))+\sqrt{\rho(y,\hat{y})}}{\|\Phi(x,\hat{y})-\Phi(x,y)\|^2}\right)\\&w \leftarrow w + \tau(\Phi(x,y) - \Phi(x,\hat{y}))\end{aligned}$$

The algorithm uses a cost function ρ. We used the function $\rho(y,\hat{y}) = 1 - F(y,\hat{y})$, where F is the soft F-measure described in Section 4.1. With this approach, the learning algorithm thus directly optimizes the measure we are interested in, i.e. the F-measure.

4 Experiments

We carried out the experiments on version 2 of the MPQA corpus (Wiebe et al., 2005), which we split into a test set (150 documents, 3,743 sentences) and a training set (541 documents, 12,010 sentences).

4.1 Evaluation Metrics

Since expression boundaries are hard to define exactly in annotation guidelines (Wiebe et al., 2005), we used soft precision and recall measures to score the quality of the system output. To derive the soft precision and recall, we first define the *span coverage* c of a span s with respect to another span s', which measures h ow well s' is covered by s:

$$c(s, s') = \frac{|s \cap s'|}{|s'|}$$

In this formula, the operator $|\cdot|$ counts tokens, and the intersection $\cap$ gives the set of tokens tha t two spans have in common. Since our evaluation takes span labels (DSE, ESE, OSE) into account, we set $c(s, s')$ to zero if the labels associated with s and s' are different.

Using the span coverage, we define the *span set coverage* C of a set of spans $\boldsymbol{S}$ with respect to a set $\boldsymbol{S'}$:

$$C(\boldsymbol{S}, \boldsymbol{S'}) = \sum_{s_j \in \boldsymbol{S}} \sum_{s'_k \in \boldsymbol{S'}} c(s_j, s'_k)$$

We now define the soft precision P and recall R of a proposed set of spans $\hat{\boldsymbol{S}}$ with respect to a gold standard set $\boldsymbol{S}$ as follows:

$$P(\boldsymbol{S}, \hat{\boldsymbol{S}}) = \frac{C(\boldsymbol{S}, \hat{\boldsymbol{S}})}{|\hat{\boldsymbol{S}}|} \quad R(\boldsymbol{S}, \hat{\boldsymbol{S}}) = \frac{C(\hat{\boldsymbol{S}}, \boldsymbol{S})}{|\boldsymbol{S}|}$$

Note that the operator $|\cdot|$ counts spans in this formula.

Conventionally, when measuring the quality of a system for an information extraction task, a predicted entity is counted as correct if it exactly matches the boundaries of a corresponding entity in the gold standard; there is thus no reward for close matches. However, since the boundaries of the spans annotated in the MPQA corpus are not strictly defined in the annotation guidelines (Wiebe et al., 2005), measuring precision and recall using exact boundary scoring will result in figures that are too low to be indicative of the usefulness of the system. Therefore, most work using this corpus instead use overlap-based precision and recall measures, where a span is counted as correctly detected if it *overlaps* with a span in the gold standard (Choi et al., 2006; Breck et al., 2007). As pointed out by Breck et al. (2007), this is problematic since it will tend to reward long spans – for instance, a span covering the whole sentence will always be counted as correct if the gold standard contains any span for that sentence.

The precision and recall measures proposed here correct the problem with overlap-based measures: If the system proposes a span covering the whole sentence, the span coverage will be low and result in a low soft precision. Note that our measures are bounded below by the exact measures and above by the overlap-based measures: replacing $c(s, s')$ with $\lfloor c(s, s') \rfloor$ gives the exact measures and replacing $c(s, s')$ with $\lceil c(s, s') \rceil$ the overlap-based measures.

To score the extraction of opinion holders, we started from the same basic approach. However, the evaluation of this task is more complex because a) we only want to give credit for holders for correctly extracted opinion expressions; b) the gold standard links opinion expressions to coreference chains rather than individual mentions of holders; c) the holder may be the writer or implicit (see 3.3). We therefore used the following method: Given a holder h linked to an expression e, we first located the expression e' in the gold standard that most closely corresponds to e, that is $e' = \arg\max_x c(x, e)$, regardless of the labels of e and e'. We then located the gold standard holder h' by finding the closest corresponding holder in the coreference chain H linked to e': $h' = \arg\max_{x \in H} c(x, h)$. If h is proposed as the writer, we score it as perfectly detected (coverage 1) if the coreference chain H contains the writer, and a full error (coverage 0) otherwise, and similar if h is implicit.

4.2 Machine Learning Methods

We compared the machine learning methods described in Section 3. In these experiments, we used a candidate set size k of 8. Table 1 shows the results of the evaluations using the precision and recall measures described above. The baseline is the result of taking the top-scoring labeling from the base model.

System	P	R	F
Baseline	63.36	46.77	53.82
Perceptron	62.84	48.13	54.51
PA	63.50	51.79	57.04

Table 1: Evaluation of reranking learning methods.

We note that the best performance was obtained using the PA algorithm. While these results are satisfactory, it is possible that they could be improved further if we would use a batch learning method such as SVM$^{\text{struct}}$ (Tsochantaridis et al., 2005) instead of the online learning methods used here.

4.3 Candidate Set Size

In any method based on reranking, it is important to study the influence of the candidate set size on the quality of the reranked output. In addition, an interesting question is what the upper bound on reranker performance is – the *oracle* performance. Table 2 shows the result of an experiment that investigates these questions. We used the reranker based on the Passive–Aggressive method in this experiment since this reranker gave the best results in the previous experiment.

	Reranked			Oracle		
k	P	R	F	P	R	F
1	63.36	46.77	53.82	63.36	46.77	53.82
2	63.70	48.17	54.86	72.66	55.18	62.72
4	63.57	49.78	55.84	79.12	62.24	69.68
8	63.50	51.79	57.04	83.72	68.14	75.13
16	63.00	52.94	57.54	86.92	72.79	79.23
32	62.15	54.50	58.07	89.18	76.76	82.51
64	61.02	55.67	58.22	91.08	80.19	85.28
128	60.22	56.45	58.27	92.63	83.00	87.55
256	59.87	57.22	58.51	94.01	85.27	89.43

Table 2: Oracle and reranker performance as a function of candidate set size.

As is common in reranking tasks, the reranker can exploit only a fraction of the potential improvement – the reduction of the F-measure error is between 10 and 15 percent of the oracle error reduction for all candidate set sizes.

The most visible effect of the reranker is that the recall is greatly improved. However, this does not seem to have an adverse effect on the precision until the candidate set size goes above 16 – in fact, the precision actually improves over the baseline for small candidate set sizes. After the size goes above 16, the recall (and the F-measure) still rises, but at the cost of decreased precision.

4.4 Syntactic and Semantic Features

We studied the impact of syntactic and semantic structural features on the performance of the reranker. Table 3 shows the result of the investigation for syntactic features. Using all the syntactic features (and no semantic features) gives an F-measure roughly 4 points above the baseline, using the PA reranker with a k of 64. We then measured the F-measure obtained when each one of the three syntactic features has been removed. It is clear that the unlexicalized syntactic path is the most important syntactic feature; the effect of the two lexicalized features seems to be negligible.

System	P	R	F
Baseline	63.36	46.77	53.82
All syntactic	62.45	53.19	57.45
No SYN PATH	64.40	48.69	55.46
No LEX PATH	62.62	53.19	57.52
No DOMINANCE	62.32	52.92	57.24

Table 3: Effect of syntactic features.

A similar result was obtained when studying the semantic features (Table 4). Removing the connecting labels feature, which is unlexicalized, has a greater effect than removing the other two semantic features, which are lexicalized.

System	P	R	F
Baseline	63.36	46.77	53.82
All semantic	61.26	53.85	57.31
No PREDICATE SL	61.28	53.81	57.30
No PRED+ARGLBL	60.96	53.61	57.05
No CONN ARGLBL	60.73	50.47	55.12

Table 4: Effect of semantic features.

4.5 Opinion Holder Extraction

Table 5 shows the performance of the opinion holder extractor. The baseline applies the holder classifier (3.3) to the opinions extracted by the base sequence labeler (3.2), without modeling any interactions between opinions. A large performance boost is then achieved simply by applying the opinion expression reranker ($k = 64$); this is simply the consequence of improved expression detection, since a correct expression is required to get credit for a holder).

However, we can improve on this by adding the holder interaction features: both the SHARED HOLDERS and HOLDER TYPES + PATH features contribute to improving the recall even further.

System	P	R	F
Baseline	57.66	45.14	50.64
Reranked expressions	52.35	52.54	52.45
SHARED HOLDERS	52.43	55.21	53.78
HTYPES + PATH	52.22	54.41	53.30
Both	52.28	55.99	54.07

Table 5: Opinion holder extraction experiments.

5 Conclusion

We have shown that features derived from grammatical and semantic role structure can be used to improve two fundamental tasks in fine-grained opinion analysis: the detection of opinionated expressions in subjectivity analysis, and the extraction of opinion holders. Our feature sets are based on interaction between opinions, which makes exact inference intractable. To overcome this issue, we used an implementation based on reranking: we first generated opinion expression sequence candidates using a simple sequence labeler similar to the approach by Breck et al. (2007). We then applied SRL-inspired opinion holder extraction classifiers, and finally a global model applying to all opinions and holders.

Our experiments show that the interaction-based models result in drastic improvements. Significantly, we see significant boosts in recall (10 points for both tasks) while the precision decreases only slightly, resulting in clear F-measure improvements. This result compares favorably with previously published results, which have been precision-oriented and scored quite low on recall.

We analyzed the impact of the syntactic and semantic features and saw that the best model is the one that makes use of both types of features. The most effective features we have found are purely structural, i.e. based on tree fragments in a syntactic or semantic tree. Features involving words did not seem to have the same impact.

There are multiple opportunities for future work in this area. An important issue that we have left open is the coreference problem for holder extraction, which has been studied by Stoyanov and Cardie (2006). Similarly, recent work has tried to incorporate complex, high-level linguistic structure such as discourse representations (Somasundaran et al., 2009); it is clear that these structures are very relevant for explaining the way humans organize their expressions of opinions rhetorically. However, theoretical depth does not necessarily guarantee practical applicability, and the challenge is as usual to find a middle ground that balances our goals: explanatory power in theory, significant performance gains in practice, computational tractability, and robustness in difficult circumstances.

6 Acknowledgements

The research described in this paper has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant 231126: LivingKnowledge – Facts, Opinions and Bias in Time, and under grant 247758: Trustworthy Eternal Systems via Evolving Software, Data and Knowledge (EternalS).

References

Bethard, Steven, Hong Yu, Ashley Thornton, Vasileios Hatzivassiloglou, and Dan Jurafsky. 2005. Extracting opinion propositions and opinion holders using syntactic and lexical cues. In Shanahan, James G., Yan Qu, and Janyce Wiebe, editors, *Computing Attitude and Affect in Text: Theory and Applications*.

Breck, Eric, Yejin Choi, and Claire Cardie. 2007. Identifying expressions of opinion in context. In *Proceedings of IJCAI-2007*, Hyderabad, India.

Choi, Yejin, Eric Breck, and Claire Cardie. 2006. Joint extraction of entities and relations for opinion recognition. In *Proceedings of EMNLP 2006*.

Collins, Michael. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8.

Crammer, Koby, Ofer Dekel, Joseph Keshet, Shai Shalev-Schwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 2006(7):551–585.

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Freund, Yoav and Robert E. Schapire. 1999. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296.

Johansson, Richard and Pierre Nugues. 2008. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *CoNLL 2008: Proceedings of the Twelfth Conference on Natural Language Learning*, pages 183–187, Manchester, United Kingdom.

Joshi, Mahesh and Carolyn Penstein-Rosé. 2009. Generalizing dependency features for opinion mining. In *Proceedings of ACL/IJCNLP 2009, Short Papers Track*.

Kim, Soo-Min and Eduard Hovy. 2006. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of ACL/COLING Workshop on Sentiment and Subjectivity in Text*.

Kobayashi, Nozomi, Kentaro Inui, and Yuji Matsumoto. 2007. Extracting aspect-evaluation and aspect-of relations in opinion mining. In *Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP-CoNLL-2007)*.

Mel'čuk, Igor A. 1988. *Dependency Syntax: Theory and Practice*. State University Press of New York, Albany.

Meyers, Adam, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. 2004. The NomBank project: An interim report. In *HLT-NAACL 2004 Workshop: Frontiers in Corpus Annotation*, pages 24–31, Boston, United States.

Moschitti, Alessandro and Roberto Basili. 2004. Complex linguistic features for text classification: A comprehensive study. In *Proceedings of ECIR*.

Palmer, Martha, Dan Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–105.

Pang, Bo, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of EMNLP*.

Ruppenhofer, Josef, Swapna Somasundaran, and Janyce Wiebe. 2008. Finding the sources and targets of subjective expressions. In *Proceedings of LREC*.

Somasundaran, Swapna, Galileo Namata, Janyce Wiebe, and Lise Getoor. 2009. Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification. In *Proceedings of EMNLP*.

Stoyanov, Veselin and Claire Cardie. 2006. Partially supervised coreference resolution for opinion summarization through structured rule learning. In *Proceedings of EMNLP 2006*.

Tjong Kim Sang, Erik F. and Jorn Veenstra. 1999. Representing text chunks. In *Proceedings of EACL99*, pages 173–179, Bergen, Norway.

Tsochantaridis, Iannis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(Sep):1453–1484.

Wiebe, Janyce, Rebecca Bruce, and Thomas O'Hara. 1999. Development and use of a gold standard data set for subjectivity classifications. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*.

Wiebe, Janyce, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3):165–210.

Wilson, Theresa, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of HLT/EMNLP 2005*.

Wu, Yuanbin, Qi Zhang, Xuanjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *Proceedings of EMNLP*.

Yu, Hong and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 129–136, Sapporo, Japan.

Unsupervised phonemic Chinese word segmentation using Adaptor Grammars

Mark Johnson
Department of Computing
Macquarie University
Mark.Johnson@mq.edu.au

Katherine Demuth
Department of Linguistics
Macquarie University
Katherine.Demuth@mq.edu.au

Abstract

Adaptor grammars are a framework for expressing and performing inference over a variety of non-parametric linguistic models. These models currently provide state-of-the-art performance on unsupervised word segmentation from phonemic representations of child-directed unsegmented English utterances. This paper investigates the applicability of these models to unsupervised word segmentation of Mandarin. We investigate a wide variety of different segmentation models, and show that the best segmentation accuracy is obtained from models that capture inter-word "collocational" dependencies. Surprisingly, enhancing the models to exploit syllable structure regularities and to capture tone information does improve overall word segmentation accuracy, perhaps because the information these elements convey is redundant when compared to the inter-word dependencies.

1 Introduction and previous work

The word-segmentation task is an abstraction of part of the problem facing a child learning its native language. Fluent speech, even the speech directed at children, doesn't come with silence or pauses delineating acoustic words the way that spaces separate orthographic words in writing systems like that of English. Instead, as most people listening to a language they don't understand can attest, words in fluent speech "run together", and a language user needs to learn how to segment utterances of the language they are learning into words. This kind of word segmentation is presumably an important first step in acquiring a language. It is scientifically interesting to know what information might be useful for word segmentation, and just how this information might be used. These scientific questions have motivated a body of research on computational models of word segmentation. Since as far as we can tell any child can learn any human language, our goal is to develop a single model that can learn to perform accurate word segmentation given input from any human language, rather than a model that specialised to perform well on a single language. This paper extends the previous work on word segmentation by investigating whether one class of models that work very well with English input also work with Chinese input. These models will permit us to study the role that syllable structure constraints and tone in Chinese might play in word segmentation.

While learners and fluent speakers undoubtedly use a wide variety of cues to perform word segmentation, computational models since Elman (1990) have tended to focus on the use of phonotactic constraints (e.g., syllable-structure constrains) and distributional information. Brent and Cartwright (1996) introduced the standard form of the word segmentation task still studied today. They extracted the orthographic representations of child-directed speech from the Bernstein-Ratner corpus (Bernstein-Ratner, 1987) and "phonologised" them by looking up each word in a pronouncing dictionary. For example, the orthographic utterance *you want to see the book* is mapped to the sequence of pronunciations *yu want tu si D6 bUk*, (the pronunciations are in an

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 528–536,
Beijing, August 2010

ASCII encoding of the International Phonetic Alphabet representation of English phonemes). The input to the learner is obtained by concatenating together the phonemic representations of each utterance's words. The learner's task is to identify the locations of the word boundaries in this sequence, and hence identify the words (up to homophony). Brent and Cartwright (1996) pointed out the importance of both distributional information and phonotactic (e.g., syllable-structure) constraints for word segmentation (see also Swingley (2005) and Fleck (2008)).

Recently there has been considerable interest in applying Bayesian inference techniques for non-parametric models to this problem. Here the term "non-parametric" does not mean that the models have no parameters, rather, it is used to distinguish these models from the usual "parametric models" that have a fixed finite vector of parameters.

Goldwater et al. (2006) introduced two non-parametric Bayesian models of word segmentation, which are discussed in more detail in (Goldwater et al., 2009). The *unigram model*, which assumes that each word is generated independently to form a sentence, turned out to be equivalent to a model originally proposed by Brent (1999). The *bigram model* improves word segmentation accuracy by modelling bigram inter-word contextual dependencies, "explaining away" inter-word dependencies that would otherwise cause the unigram model to under-segment. Mochihashi et al. (2009) showed that segmentation accuracy could be improved by using a more sophisticated "base distribution" and a dynamic programming sampling algorithm very similar to the one used with the adaptor grammars below. They also applied their algorithm to Japanese and Chinese word segmentation, albeit from orthographic rather than phonemic forms, so unfortunately their results are not comparable with ours.

Johnson et al. (2007) introduced *adaptor grammars* as a grammar-based framework for expressing a variety of non-parametric models, and provided a dynamic programming Markov Chain Monte Carlo (MCMC) sampling algorithm for performing Bayesian inference on these models. For example, the unigram model can be expressed as a simple adaptor grammar as shown below, and the generic adaptor grammar inference procedure provides a dynamic programming sampling algorithm for this model. Johnson (2008b) showed how a variety of different word segmentation models can be expressed as adaptor grammars, and Johnson and Goldwater (2009) described a number of extensions and specialisations to the adaptor grammar framework that improve inference speed and accuracy (we use these techniques in our work below).

Previous work on unsupervised word segmentation from phonemic input has tended to concentrate on English. However, presumably children the world over segment their first language input in the same (innately-specified) way, so a correct procedure should work for all possible human languages. However, as far as we are aware there has been relatively little work on word segmentation from phonemic input except on English. Johnson (2008a) investigated whether the adaptor grammars models that do very well on English also apply to Sesotho (a Bantu language spoken in southern Africa with rich agglutinating morphology). He found that the models in general do very poorly (presumably because the adaptor grammars used cannot model the complex morphology found in Sesotho) and that the best segmentation accuracy was considerably worse than that obtained for English, even when that model incorporated some Bantu-specific information about morphology. Of course it may also be that the Sesotho and English corpora are not really comparable: the Bernstein-Ratner corpus that Brent and other researchers have used for English was spoken to pre-linguistic 1-year olds, while most non-English corpora are of child-directed speech to older children who are capable of talking back, and hence these corpora are presumably more complex. We discuss this issue in more detail in section 4 below.

2 A Chinese word segmentation corpus

Our goal here is to prepare a Chinese corpus of child-directed speech that parallels the English one used by Brent and other researchers. That corpus was in broad phonemic form, obtained by looking each word up in a pronouncing dictionary. Here instead we make use of a corpus in Pinyin format, which we translate into a broad phonemic IPA format using the freely-available Pinyin-to-IPA translation program "Pinyin to IPA Conversion Tools" version 2.1 available on http://sourceforge.net/projects/py2ipa.

We used the "Beijing" corpus (Tardif, 1993) available from the publicly-distributed Childes collection of corpora (MacWhinney and Snow, 1985). We are interested in child-directed speech (rather than children's speech), so we removed all utterances from participants with an Id containing "Child". (Tardif (1993) points out that Chinese-speaking children typically have a much richer social environment involving multiple adult care-givers than middle-class English-speaking children do, so we cannot simply collect only the mother's utterances, as was done for the English corpus). We also ignored all utterances with codes $INTERJ, $UNINT, $VOC and $PRMPT, as these are not always linguistic utterances. In addition, we deleted all words that could not be analysed as a sequence of syllables, such as "xxx" and "hmm", and also deleted "cluck". The first few utterances of the corpus in Pinyin format are:

zen3me gei3 ta1 bei1 shang4 lai2 (1.) ?
ta1: (.) a1yi2 gei3 de (.) ta1 gei3 de .
hen3 jian3dan1 .

We then fed these into the Pinyin-to-IPA translation program, producing output of the following format:

tsən^{214}mɤ kei^{214} t^{h}a^{55} pei^{55} ʂɑŋ^{51} lai^{35}
t^{h}a^{55} a^{55}i^{35} kei^{214} tɤ t^{h}a^{55} kei^{214} tɤ
xən^{214} tɕiɛn^{214}tan^{55}

In the IPA format, the superscript indices indicate the tone patterns associated with syllables; these appear at the end of each syllable, as is standard. While we believe there are good linguistic reasons to analyse tones as associated with syllables, we moved all the tones so they immediately followed the final vowel in each syllable. We did this because we thought that locating tones after the syllable-final consonant might give our models a strong cue as to the location of syllable boundaries, and since words often end at syllable boundaries, this would make the word segmentation problem artificially easier. (Our models take a sequence of symbols as input, so the tones must be located somewhere in the sequence. However, the linguistically "correct" solution would probably be to extend the models so they could process input in an auto-segmental format (Goldsmith, 1990) where tones would be on a separate tier and unordered with respect to the segments within a syllable.)

In order to evaluate the importance of tone for our word-segmentation models we also constructed a version of our corpus in which all tones were removed. We present results for all of our models on two versions of the corpus, one that contains tones following the vowels, and another that contains no tones at all. These two corpora constitute the "gold standard" against which our word segmentation models will be evaluated. These corpora contain 50,118 utterances, consisting of 187,533 word tokens.

The training data provided to the word segmentation models is obtained by segmenting the gold data at all possible boundary locations. Consonant clusters, diphthongs and tones (if present) are treated as single units, so the training data appears as follows:

ts ə ^{214} n m ɤ k e i ^{214} t^{h} a ^{55} p e i ^{55} ʂ ɑ ^{51} ŋ l ai ^{35}
t^{h} a ^{55} a ^{55} i ^{35} k e i ^{214} t ɤ t^{h} a ^{55} k e i ^{214} t ɤ
x ə ^{214} n tɕ iɛ ^{214} n t a ^{55} n

The task of a word-segmentation model is to identify which of these possible boundary locations correspond to actual word boundaries. The training corpus without tones contains 531,384 segments, while the training corpus with tones contains 712,318 segments.

3 Adaptor grammars for word segmentation

Adaptor grammars were first introduced by Johnson et al. (2007) as a grammar-based framework for specifying hierarchical non-parametric Bayesian models, and Johnson and Goldwater (2009) describes a number of implementation details that significantly improve performance; the interested reader should consult those papers for a full technical introduction. Johnson (2008b) proposed a number of adaptor grammars for English word segmentation, which we review and minimally modify here so they can perform Chinese word segmentation as well. In section 4 we evaluate these adaptor grammars on the Chinese corpus just described.

The grammars vary along two orthogonal dimensions, which correspond to the kinds of generalisations that the model can learn. The simplest grammar is the unigram adaptor grammar, which generates an utterance as an i.i.d. sequences of words, where each word is a sequence of phonemes. The collocation adaptor grammars capture dependencies above the word level by generating collocations, or groups of words, as memoized units. The syllable adaptor grammars capture dependencies below the word level by generating words as sequences of syllables rather than phonemes.

3.1 Unigram adaptor grammars

In order to motivate adaptor grammars as an extension to Probabilistic Context-Free Grammars (PCFGs), consider an attempt to perform unsupervised word segmentation with a PCFG containing the following rules (ignore the underlining of the Word non-terminal for now).

$$
\begin{array}{l}
\text{Words} \rightarrow \text{Words } \underline{\text{Word}} \\
\text{Words} \rightarrow \underline{\text{Word}} \\
\underline{\text{Word}} \rightarrow \text{Phons} \\
\text{Phons} \rightarrow \text{Phon} \\
\text{Phons} \rightarrow \text{Phons Phon} \\
\text{Phons} \rightarrow \text{Phons Tone} \\
\text{Phon} \rightarrow \text{ai} \mid \text{o} \mid ... \mid \text{ʂ} \mid \text{tʂ}^{h} \mid ... \\
\text{Tone} \rightarrow 35 \mid 55 \mid 214 \mid ...
\end{array}
\quad (1)
$$

In this grammar, Phon expands to all the phonemes appearing in the phonemic training data, and Tone expands to all of the tone patterns. (In this and all of the other grammars in this paper, the start symbol is the non-terminal symbol of the first rule in the grammar. This grammar, like all others in this paper, is crafted so that a Word subtree can never begin with a Tone, so the presence of tones does not make the segmentation problem harder).

The trees generated by this grammar are sufficiently expressive to *represent* any possible segmentation of any sequence of phonemes into words (including the true segmentation); a typical segmentation is shown in Figure 1. However,

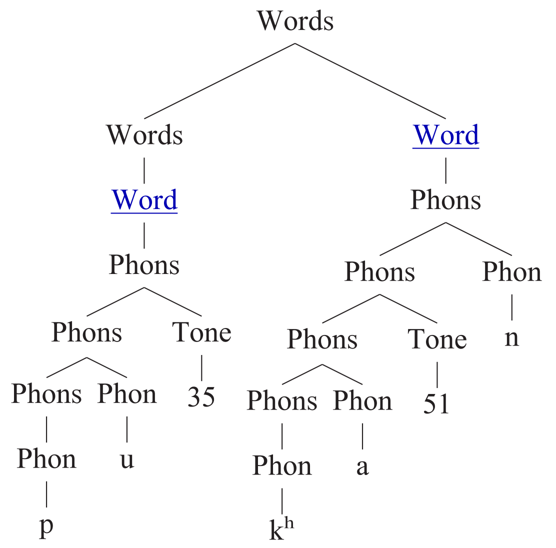


Figure 1: A parse tree generated by the unigram grammar, where adapted and non-adapted non-terminals are shown. It depicts a possible segmentation of p u ^{35} k^h a ^{51} n.

it should also be clear that no matter how we vary the probabilities on the rules of this grammar, *the grammar itself cannot encode the subset of trees that correspond to words of the language*. In order to do this, a model would need to memorise the probabilities of entire Word subtrees, since these are the units that correspond to individual words, but this PCFG simply is not expressive enough to do this.

Adaptor grammars learn the probabilities of subtrees in just this way. An adaptor grammar is specified via a set of rules or productions, just like a CFG, and the set of trees that an adaptor grammar generates is exactly the same as the CFG with those rules.

However, an adaptor grammar defines probability distributions over trees in a completely different fashion to a PCFG: for simplicity we focus here on the sampling or predictive distribution, which defines the probability of generating an entire corpus of trees. In a PCFG, the probability of each non-terminal expanding using a given rule is determined by the probability of that rule, and is independent of the expansions of the other non-terminals in the tree. In an adaptor grammar a subset of the non-terminals are designated as *adapted*. We indicate adapted non-terminals by underlining them, so <u>Word</u> is the only adapted non-terminal in (1). Unadapted non-terminals expand just as in a PCFG: a production is chosen according to the production probabilities. An adapted non-terminal can expand in two different ways. With probability proportional to $n(t) - a_A$ an adapted non-terminal A expands to a tree t rooted in A that has been previously generated, while with probability proportional to $m(A)a_A + b_A$ the adapted non-terminal A expands using some grammar rule, just as in a PCFG. Here $n(t)$ is the number of times tree t has been previously generated, $m(A)$ is the number of trees rooted in A that have been previously generated using grammar rules, and $0 \leqslant a_A \leqslant 1$ and $b_A > 0$ are adjustable parameters associated with the adapted non-terminal A.

Technically, this is known as a *Pitman-Yor Process* (PYP) with *concentration parameters* a_A and b_A, where the PCFG rules define the *base distribution* of the process. (The PYP is a generalisation of the Chinese Restaurant Process (CRP); a CRP is a PYP with parameter $a = 0$). Rather than setting the concentration parameters by hand (there are two for each adapted non-terminal in the grammar) we follow Johnson and Goldwater (2009) and put uniform Beta and vague Gamma priors on each of these parameters, and use sampling to explore their posterior values.

Because the probability of selecting a tree t is proportional to $n(t)$, an adaptor grammar is a kind of "rich-get-richer" process that generates power-law distributions. Depending on the values of a_A and b_A, most of the probability mass can wind up concentrated on just a few trees. An adaptor grammar is a kind of "cache" model, in which previously generated subtrees are stored and more likely to be reused in later sentences. That is, while an adapted non-terminal A can expand to any tree rooted in A that can be constructed with the grammar rules, in practice it is increasingly likely to reuse the same trees over and over again. It can be viewed as a kind of tree substitution grammar (Joshi, 2003), but where the tree fragments (as well as their probabilities) are learnt from the data.

The unigram grammar is the simplest of the word segmentation models we investigate in this paper (it is equivalent to the unigram model investigated in Goldwater et al. (2009)). Because the grammars we present below rapidly become long and complicated to read if each grammar rule is explicitly stated, we adopt the following conventions. We use regular expressions to abbreviate our grammars, with the understanding that the regular expressions are always expanded produce a left-recursive structure. For example, the unigram grammar in (1) is abbreviated as:

$$\begin{array}{l} \text{Words} \rightarrow \underline{\text{Word}}^+ \\ \underline{\text{Word}} \rightarrow \text{Phon (Phon} \mid \text{Tone)}^\star \\ \text{Phon} \rightarrow \text{ai} \mid \text{o} \mid \ldots \mid \text{ʂ} \mid \text{tʂ}^{\text{h}} \mid \ldots \\ \text{Tone} \rightarrow 35 \mid 55 \mid 214 \mid \ldots \end{array} \qquad (2)$$

3.2 Collocation adaptor grammars

Goldwater et al. (2006) and Goldwater et al. (2009) demonstrated the importance of contextual dependencies for word segmentation, and proposed a bigram model in order to capture some of these. It turns out that while the bigram model cannot be expressed as an adaptor grammar, a *collocation model*, which captures similar kinds of contextual dependencies, can be expressed as an adaptor grammar (Johnson et al., 2007). In a collocation grammar there are two different adapted non-terminals; <u>Word</u> and <u>Colloc</u>; <u>Word</u> expands exactly as in the unigram grammar (2), so it is not repeated here.

$$\begin{array}{l} \text{Collocs} \rightarrow \underline{\text{Colloc}}^+ \\ \underline{\text{Colloc}} \rightarrow \text{Words} \\ \text{Words} \rightarrow \underline{\text{Word}}^+ \end{array} \qquad (3)$$

A collocation adaptor grammar caches both words and collocations (which are sequences of words). An utterance is generated by generating one or more collocations. The PYP associated with collocations either regenerates a previously generated collocation or else generates a "fresh" collocation by generating a sequence of words according to the PYP model explained above.

The idea of aggregating words into collocations can be reapplied at a more abstract level by aggregating collocations into "super-collocations", which are sequences of collocations. This involves adding the following additional rules to the grammar in (3):

$$\begin{aligned} \text{Colloc2s} &\rightarrow \underline{\text{Colloc2}}^{+} \\ \underline{\text{Colloc2}} &\rightarrow \text{Collocs}^{+} \end{aligned} \tag{4}$$

There are three PYPs in a grammar with 2 levels of collocations, arranged in a strict Bayesian hierarchy. It should be clear that this process can be repeated indefinitely; we investigate grammars with up to three levels of collocations below. (It should be possible to use Bayesian techniques to learn the appropriate number of levels in the hierarchy, but we leave this for future work).

3.3 Syllable structure adaptor grammars

Brent and Cartwright (1996) and others emphasise the role that syllable-structure and other phonotactic constraints might play in word segmentation. Johnson (2008b) pointed out that adaptor grammars can learn at least some of these kinds of generalisations. It's not unreasonable to assume that language learners can learn to group phonemes into syllables, and that they can exploit this syllabic structure to perform word segmentation. The syllable-structure grammars we describe below assume that word boundaries are always aligned with syllable boundaries; this is not universally true, but it is reliable enough to dramatically improve unsupervised word segmentation in English.

There is considerable cross-linguistic variation in the syllable-structure and phonotactic constraints operative in the languages of the world, so we'd like to avoid "building in" language-specific constraints into our model. We therefore make the relatively conservative assumption that the child can distinguish vowels from consonants, and that the child knows that syllables consist of Onsets, Nuclei and Codas, that Onsets and Codas consist of arbitrary sequences of consonants while Nuclei are arbitrary sequences of vowels and tones, and that Onsets and Codas are optional. Notice that syllable structure in both English and Chinese is considerably more constrained than this; we use this simple model here because it has proved successful for English word segmentation.

The syllable-structure adaptor grammars replace the rules expanding Words with the following rules:

$$\begin{aligned} \underline{\text{Word}} &\rightarrow \text{Syll} \\ \underline{\text{Word}} &\rightarrow \text{Syll Syll} \\ \underline{\text{Word}} &\rightarrow \text{Syll Syll Syll} \\ \underline{\text{Word}} &\rightarrow \text{Syll Syll Syll Syll} \\ \text{Syll} &\rightarrow (\underline{\text{Onset}})^{?}\ \underline{\text{Rhy}} \\ \underline{\text{Onset}} &\rightarrow \text{C}^{+} \\ \underline{\text{Rhy}} &\rightarrow \underline{\text{Nucleus}}\ (\underline{\text{Coda}})^{?} \\ \underline{\text{Nucleus}} &\rightarrow \text{V}\ (\text{V} \mid \text{Tone})^{\star} \\ \underline{\text{Coda}} &\rightarrow \text{C}^{+} \\ \text{C} &\rightarrow \text{ʂ} \mid \text{tʂ}^{\text{h}} \mid \ldots \\ \text{V} &\rightarrow \text{ai} \mid \text{o} \mid \ldots \end{aligned} \tag{5}$$

In these rules the superscript "?" indicates optionality. We used the relatively cumbersome mechanism of enumerating each possible number of syllables per word (we permit words to consist of from 1 to 4 syllables, although ideally this number would not be hard-wired into the grammar) because a relatively trivial modification of this grammar can distinguish word-initial and word-final consonant clusters from word-internal clusters. Johnson (2008b) demonstrated that this significantly improves English word segmentation accuracy. We do not expect this to improve Chinese word segmentation because Chinese clusters do not vary depending on their location within the word, but it will be interesting to see if the additional cluster flexibility that is useful for English segmentation hurts Chinese segmentation.

In this version of the syllable-structure grammar, we replace the Word rules in the syllable adaptor grammar with the following:

$$\begin{aligned} \underline{\text{Word}} &\rightarrow \text{SyllIF} \\ \underline{\text{Word}} &\rightarrow \text{SyllI SyllF} \\ \underline{\text{Word}} &\rightarrow \text{SyllI Syll SyllF} \\ \underline{\text{Word}} &\rightarrow \text{SyllI Syll Syll SyllF} \end{aligned} \tag{6}$$

and add the following rules expanding the new kinds of syllables to the rules in (5).

$$\begin{aligned} \text{SyllIF} &\rightarrow (\underline{\text{OnsetI}})^{?}\ \underline{\text{RhyF}} \\ \text{SyllI} &\rightarrow (\underline{\text{OnsetI}})^{?}\ \underline{\text{Rhy}} \\ \text{SyllF} &\rightarrow (\underline{\text{OnsetI}})^{?}\ \underline{\text{RhyF}} \\ \text{Syll} &\rightarrow (\underline{\text{Onset}})^{?}\ \underline{\text{Rhy}} \\ \underline{\text{OnsetI}} &\rightarrow \text{C}^{+} \\ \underline{\text{RhyF}} &\rightarrow \underline{\text{Nucleus}}\ (\underline{\text{CodaF}})^{?} \\ \underline{\text{CodaF}} &\rightarrow \text{C}^{+} \end{aligned} \tag{7}$$

	Syllables		
	None	General	Specialised
Unigram	0.57	0.50	0.50
Colloc	0.69	0.67	0.67
Colloc^2	0.72	0.75	0.75
Colloc^3	*0.64*	**0.77**	**0.77**

Table 1: F-score accuracies of word segmentations produced by the adaptor grammar models on the Chinese corpus *with tones*.

	Syllables		
	None	General	Specialised
Unigram	0.56	0.46	0.46
Colloc	0.70	0.65	0.65
Colloc^2	0.74	0.74	0.73
Colloc^3	0.75	0.76	**0.77**

Table 2: F-score accuracies of word segmentations produced by the adaptor grammar models on the Chinese corpus *without tones*.

These rules distinguish syllable onsets in word-initial position and syllable codas in word-final position; the standard adaptor grammar machinery will then learn distributions over onsets and codas in these positions that possibly differ from those in word-internal positions.

4 Results on Chinese word segmentation

The previous section described two dimensions along which adaptor grammars for word segmentation can independently vary. Above the Word level, there can be from zero to three levels of collocations, yielding four different values for this dimension. Below the Word level, phonemes can either be treated as independent entities, or else they can be grouped into onset, nuclei and coda clusters, and these can vary depending on where they appear within a word. Thus there are three different values for the syllable dimension, so there are twelve different adaptor grammars overall. In addition, we ran all of these grammars on two versions of the corpus, one with tones and one without tones, so we report results for 24 different runs here.

The adaptor grammar inference procedure we used is the one described in Johnson and Goldwater (2009). We ran 1,000 iterations of 8 MCMC chains for each run, and we discarded all but last 200 iterations in order to "burn-in" the sampler. The segmentation we predict is the one that occurs the most frequently in the samples that were not discarded. As is standard, we evaluate the models in terms of token f-score; the results are presented in Tables 1 and 2.

In these tables, "None" indicates that the grammar does not model syllable structure, "General" indicates that the grammar does not distinguish word-peripheral from word-internal clusters, while "Specialised" indicates that it does. "Unigram" indicates that the grammar does not model collocational structure, otherwise the superscript indicates the number of collocational levels that the grammar captures.

Broadly speaking, the results are consistent with the English word segmentation results using adaptor grammars presented by Johnson (2008b). The unigram grammar segmentation accuracy is similar to that obtained for English, but the results for the other models are lower than the results for the corresponding adaptor grammars on English.

We see a general improvement in segmentation accuracy as the number of collocation levels increases, just as for English. However, we do not see any general improvements associated with modelling syllables; indeed, it seems modelling syllables causes accuracy to decrease unless collocational structure is also modelled. This is somewhat surprising, as Chinese has a very regular syllabic structure. It is not surprising that distinguishing word-peripheral and word-medial clusters does not improve segmentation accuracy, as Chinese does not distinguish these kinds of clusters. There is also no sign of the "synergies" when modelling collocations and syllables together that Johnson (2008b) reported.

It is also surprising that tones seem to make little difference to the segmentation accuracy, since they are crucial for disambiguating lexical items. The segmentation accuracy of the models that capture little or no inter-word dependencies (e.g., Unigram, Colloc) improved slightly when the input contains tones, but the best-performing models that capture a more complex set of inter-word dependencies do equally well on the corpus without tones as they do on the corpus with tones. Because these models capture rich inter-word context (they model three levels of collocational structure), it is possible that this context provides sufficient information to segment words even in the absence of tone information, i.e., the tonal information is redundant given the richer inter-word dependencies that these models capture. It is also possible that word segmentation may simply require less information than lexical disambiguation.

One surprising result is the relatively poor performance of the Colloc^3 model without syllables but with tones; we have no explanation for this. However, all 8 of the MCMC chains in this run produced lower f-scores, so it unlikely to be simply a random fluctuation produced by a single outlier.

Note that one should be cautious when comparing the absolute f-scores from these experiments with those of the English study, as the English and Chinese corpora differ in many ways. As Tardif (1993) (the creator of the Chinese corpus) emphasises, this corpus was collected in a much more diverse linguistic environment with child-directed speech from multiple caregivers. The children involved in the Chinese corpus were also older than the children in the English corpus, which may also have affected the nature of the corpus.

5 Conclusion

This paper applied adaptor grammar models of phonemic word segmentation originally developed for English to Chinese data. While the Chinese data was prepared in a very different way to the English data, the adaptor grammars used to perform Chinese word segmentation were very similar to those used for the English word segmentation. They also achieved quite respectable f-score accuracies, which suggests that the same models can do well on both languages.

One puzzling result is that incorporating syllable structure phonotactic constraints, which enhances English word segmentation accuracy considerably, doesn't seem to improve Chinese word segmentation to a similar extent. This may reflect the fact that the word segmentation adaptor grammars were originally designed and tuned for English, and perhaps differently formulated syllable-structure constraints would work well for Chinese. But even if one can "tune" the adaptor grammars to improve performance on Chinese, the challenge is doing this in a way that improves performance on all languages, rather than just one.

Acknowledgments

The authors would like to thank the US NSF for support for this research, which was begun while they were on the faculty at Brown University in the USA. The NSF supported this work through NSF awards 0544127 and 0631667 to Mark Johnson and Katherine Demuth.

The adaptor grammar software is freely available for download from http://web.science.mq.edu.au/~mjohnson, and the Chinese data was obtained from the Childes archive.

References

Bernstein-Ratner, N. 1987. The phonology of parent-child speech. In Nelson, K. and A. van Kleeck, editors, *Children's Language*, volume 6. Erlbaum, Hillsdale, NJ.

Brent, M. and T. Cartwright. 1996. Distributional regularity and phonotactic constraints are useful for segmentation. *Cognition*, 61:93–125.

Brent, M. 1999. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34:71–105.

Elman, Jeffrey. 1990. Finding structure in time. *Cognitive Science*, 14:197–211.

Fleck, Margaret M. 2008. Lexicalized phonotactic word segmentation. In *Proceedings of ACL-08: HLT*, pages 130–138, Columbus, Ohio, June. Association for Computational Linguistics.

Goldsmith, John A. 1990. *Autosegmental and Metrical Phonology*. Basil Blackwell, Oxford, England.

Goldwater, Sharon, Thomas L. Griffiths, and Mark Johnson. 2006. Contextual dependencies in unsupervised word segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 673–680, Sydney, Australia. Association for Computational Linguistics.

Goldwater, Sharon, Thomas L. Griffiths, and Mark Johnson. 2009. A Bayesian framework for word segmentation: Exploring the effects of context. *Cognition*, 112(1):21 – 54.

Johnson, Mark and Sharon Goldwater. 2009. Improving nonparameteric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 317–325, Boulder, Colorado, June. Association for Computational Linguistics.

Johnson, Mark, Thomas L. Griffiths, and Sharon Goldwater. 2007. Adaptor Grammars: A framework for specifying compositional nonparametric Bayesian models. In Schölkopf, B., J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 641–648. MIT Press, Cambridge, MA.

Johnson, Mark. 2008a. Unsupervised word segmentation for Sesotho using adaptor grammars. In *Proceedings of the Tenth Meeting of ACL Special Interest Group on Computational Morphology and Phonology*, pages 20–27, Columbus, Ohio, June. Association for Computational Linguistics.

Johnson, Mark. 2008b. Using adaptor grammars to identifying synergies in the unsupervised acquisition of linguistic structure. In *Proceedings of the 46th Annual Meeting of the Association of Computational Linguistics*, Columbus, Ohio. Association for Computational Linguistics.

Joshi, Aravind. 2003. Tree adjoining grammars. In Mikkov, Ruslan, editor, *The Oxford Handbook of Computational Linguistics*, pages 483–501. Oxford University Press, Oxford, England.

MacWhinney, Brian and Catherine Snow. 1985. The child language data exchange system. *Journal of Child Language*, 12:271–296.

Mochihashi, Daichi, Takeshi Yamada, and Naonori Ueda. 2009. Bayesian unsupervised word segmentation with nested Pitman-Yor language modeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 100–108, Suntec, Singapore, August. Association for Computational Linguistics.

Swingley, Dan. 2005. Statistical clustering and the contents of the infant vocabulary. *Cognitive Psychology*, 50:86–132.

Tardif, Twila. 1993. *Adult-to-child speech and language acquisition in Mandarin Chinese*. Ph.D. thesis, Yale University.

Data-Driven Parsing with Probabilistic Linear Context-Free Rewriting Systems

Laura Kallmeyer and **Wolfgang Maier**
SFB 833, University of Tübingen
{lk,wmaier}@sfs.uni-tuebingen.de

Abstract

This paper presents a first efficient implementation of a weighted deductive CYK parser for Probabilistic Linear Context-Free Rewriting Systems (PLCFRS), together with context-summary estimates for parse items used to speed up parsing. LCFRS, an extension of CFG, can describe discontinuities both in constituency and dependency structures in a straightforward way and is therefore a natural candidate to be used for data-driven parsing. We evaluate our parser with a grammar extracted from the German NeGra treebank. Our experiments show that data-driven LCFRS parsing is feasible with a reasonable speed and yields output of competitive quality.

1 Introduction

Data-driven parsing has largely been dominated by Probabilistic Context-Free Grammar (PCFG). The use of PCFG is tied to the annotation principles of popular treebanks, such as the Penn Treebank (PTB) (Marcus et al., 1994), which are used as a data source for grammar extraction. Their annotation generally relies on the use of trees without crossing branches, augmented with a mechanism that accounts for non-local dependencies. In the PTB, e.g., labeling conventions and trace nodes are used which establish additional implicit edges in the tree beyond the overt phrase structure. In contrast, some other treebanks, such as the German NeGra and TIGER treebanks allow annotation with crossing branches (Skut et al., 1997). Non-local dependencies can then be expressed directly by grouping all dependent elements under a single node.

However, given the expressivity restrictions of PCFG, work on data-driven parsing has mostly excluded non-local dependencies. When using treebanks with PTB-like annotation, labeling conventions and trace nodes are often discarded, while in NeGra, resp. TIGER, tree transformations are applied which resolve the crossing branches (Kübler, 2005; Boyd, 2007, e.g.). Especially for these treebanks, such a transformation is questionable, since it is non-reversible and implies information loss.

Some research has gone into incorporating non-local information into data-driven parsing. Levy and Manning (2004) distinguish three approaches: 1. Non-local information can be incorporated directly into the PCFG model (Collins, 1999), or can be reconstructed in a post-processing step after PCFG parsing (Johnson, 2002; Levy and Manning, 2004). 2. Non-local information can be incorporated into complex labels (Hockenmaier, 2003). 3. A formalism can be used which accommodates the direct encoding of non-local information (Plaehn, 2004). This paper pursues the third approach.

Our work is motivated by the following recent developments: Linear Context-Free Rewriting Systems (LCFRS) (Vijay-Shanker et al., 1987) have been established as a candidate for modeling both discontinuous constituents and non-projective dependency trees as they occur in treebanks (Kuhlmann and Satta, 2009; Maier and Lichte, 2009). LCFRS extend CFG such that non-terminals can span tuples of possibly non-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 537–545,
Beijing, August 2010

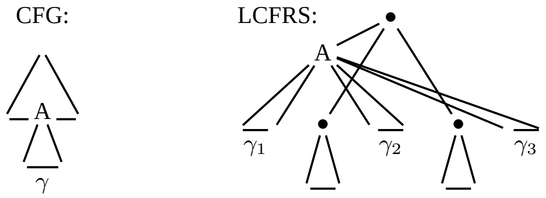


Figure 1: Different domains of locality

adjacent strings (see Fig. 1). PCFG techniques, such as Best-First Parsing (Charniak and Caraballo, 1998), Weighted Deductive Parsing (Nederhof, 2003) and A* parsing (Klein and Manning, 2003a), can be transferred to LCFRS. Finally, German has attracted the interest of the parsing community due to the challenges arising from its frequent discontinuous constituents (Kübler and Penn, 2008).

We bring together these developments by presenting a parser for probabilistic LCFRS. While parsers for subclasses of PLCFRS have been presented before (Kato et al., 2006), to our knowledge, our parser is the first for the entire class of PLCFRS. We have already presented an application of the parser on constituency and dependency treebanks together with an extensive evaluation (Maier, 2010; Maier and Kallmeyer, 2010). This article is mainly dedicated to the presentation of several methods for context summary estimation of parse items, and to an experimental evaluation of their usefulness. The estimates either act as figures-of-merit in a best-first parsing context or as estimates for A* parsing. Our evaluation shows that while our parser achieves a reasonable speed already without estimates, the estimates lead to a great reduction of the number of produced items, all while preserving the output quality.

Sect. 2 and 3 of the paper introduce probabilistic LCFRS and the parsing algorithm. Sect. 4 presents different context summary estimates. In Sect. 5, the implementation and evaluation of the work is discussed.

2 Probabilistic LCFRS

LCFRS are an extension of CFG where the non-terminals can span not only single strings but, instead, tuples of strings. We will notate LCFRS with the syntax of *simple Range Concatenation Grammars* (SRCG) (Boullier, 1998), a formalism that is equivalent to LCFRS.

A LCFRS (Vijay-Shanker et al., 1987) is a tuple $\langle N, T, V, P, S\rangle$ where a) N is a finite set of non-terminals with a function dim: $N \to \mathbb{N}$ that determines the *fan-out* of each $A \in N$; b) T and V are disjoint finite sets of terminals and variables; c) $S \in N$ is the start symbol with $dim(S) = 1$; d) P is a finite set of rules

$$A(\alpha_1, \dots, \alpha_{dim(A)}) \to A_1(X_1^{(1)}, \dots, X_{dim(A_1)}^{(1)}) \cdots A_m(X_1^{(m)}, \dots, X_{dim(A_m)}^{(m)})$$

for $m \geq 0$ where $A, A_1, \dots, A_m \in N$, $X_j^{(i)} \in V$ for $1 \leq i \leq m, 1 \leq j \leq dim(A_i)$ and $\alpha_i \in (T \cup V)^*$ for $1 \leq i \leq dim(A)$. For all $r \in P$, it holds that every variable X occurring in r occurs exactly once in the left-hand side (LHS) and exactly once in the right-hand side (RHS).

A rewriting rule describes how the yield of the LHS non-terminal can be computed from the yields of the RHS non-terminals. The rules $A(ab, cd) \to \varepsilon$ and $A(aXb, cYd) \to A(X, Y)$ for instance specify that 1. $\langle ab, cd\rangle$ is in the yield of A and 2. one can compute a new tuple in the yield of A from an already existing one by wrapping a and b around the first component and c and d around the second.

For every $A \in N$ in a LCFRS G, we define the yield of A, $yield(A)$ as follows:

a) For every $A(\vec{\alpha}) \to \varepsilon$, $\vec{\alpha} \in yield(A)$;
b) For every rule
$$A(\alpha_1, \dots, \alpha_{dim(A)}) \to A_1(X_1^{(1)}, \dots, X_{dim(A_1)}^{(1)}) \cdots A_m(X_1^{(m)}, \dots, X_{dim(A_m)}^{(m)})$$
and all $\vec{\tau_i} \in yield(A_i)$ for $1 \leq i \leq m$, $\langle f(\alpha_1), \dots, f(\alpha_{dim(A)})\rangle \in yield(A)$ where f is defined as follows: (i) $f(t) = t$ for all $t \in T$, (ii) $f(X_j^{(i)}) = \vec{\tau_i}(j)$ for all $1 \leq i \leq m, 1 \leq j \leq dim(A_i)$ and (iii) $f(xy) = f(x)f(y)$ for all $x, y \in (T \cup V)^+$. f is the *composition function* of the rule.
c) Nothing else is in $yield(A)$.

The language is then $\{w \mid \langle w\rangle \in yield(S)\}$.

The *fan-out* of an LCFRS G is the maximal fan-out of all non-terminals in G. Furthermore, the RHS length of a rewriting rules $r \in P$ is called the *rank* of r and the maximal rank of all rules in P is called the *rank* of G. We call a LCFRS *ordered* if for every $r \in P$ and every RHS non-terminal A in r and each pair X_1, X_2 of arguments of A in the RHS of r, X_1 precedes X_2 in the RHS iff X_1 precedes X_2 in the LHS.

A *probabilistic LCFRS* (PLCFRS) (Kato et al., 2006) is a tuple $\langle N, T, V, P, S, p\rangle$ such that $\langle N, T, V, P, S\rangle$ is a LCFRS and $p : P \to [0..1]$ a function such that for all $A \in N$: $\Sigma_{A(\vec{x})\to\vec{\Phi}\in P} p(A(\vec{x}) \to \vec{\Phi}) = 1$.

3 The CYK Parser

We use a probabilistic version of the CYK parser from (Seki et al., 1991), applying techniques of weighted deductive parsing (Nederhof, 2003).

LCFRS can be binarized (Gómez-Rodríguez et al., 2009) and ε-components in the LHS of rules can be removed (Boullier, 1998). We can therefore assume that all rules are of rank 2 and do not contain ε components in their LHS. Furthermore, we assume POS tagging to be done before parsing. POS tags are non-terminals of fan-out 1. The rules are then either of the form $A(a) \to \varepsilon$ with A a POS tag and $a \in T$ or of the form $A(\vec{\alpha}) \to B(\vec{x})$ or $A(\vec{\alpha}) \to B(\vec{x})C(\vec{y})$ where $\vec{\alpha} \in (V^+)^{dim(A)}$, i.e., only the rules for POS tags contain terminals in their LHSs.

For every $w \in T^*$, where $w = w_1 \dots w_n$ with $w_i \in T$ for $1 \leq i \leq n$, we define: $Pos(w) := \{0, \dots, n\}$. A pair $\langle l, r\rangle \in Pos(w) \times Pos(w)$ with $l \leq r$ is a *range* in w. Its *yield* $\langle l, r\rangle(w)$ is the string $w_{l+1} \dots w_r$. The yield $\vec{\rho}(w)$ of a vector of ranges $\vec{\rho}$ is the vector of the yields of the single ranges. For two ranges $\rho_1 = \langle l_1, r_1\rangle, \rho_2 = \langle l_2, r_2\rangle$: if $r_1 = l_2$, then $\rho_1 \cdot \rho_2 = \langle l_1, r_2\rangle$; otherwise $\rho_1 \cdot \rho_2$ is undefined.

For a given rule $p : A(\alpha_1, \dots, \alpha_{dim(A)}) \to B(X_1, \dots, X_{dim(B)})C(Y_1, \dots, X_{dim(C)})$ we now extend the composition function f to ranges, given an input w: for all range vectors $\vec{\rho_B}$ and $\vec{\rho_C}$ of dimensions $dim(B)$ and $dim(C)$ respectively, $f_r(\vec{\rho_B}, \vec{\rho_C}) = \langle g(\alpha_1), \dots, g(\alpha_{dim(A)})\rangle$ is defined as follows: $g(X_i) = \vec{\rho_B}(i)$ for all $1 \leq i \leq dim(B)$, $g(Y_i) = \vec{\rho_C}(i)$ for all $1 \leq i \leq dim(C)$ and $g(xy) = g(x) \cdot g(y)$ for all $x, y \in V^+$. $p : A(f_r(\vec{\rho_B}, \vec{\rho_C})) \to B(\vec{\rho_B})C(\vec{\rho_C})$ is then called an *instantiated rule*.

For a given input w, our items have the form $[A, \vec{\rho}]$ where $A \in N$, $\vec{\rho} \in (Pos(w) \times Pos(w))^{dim(A)}$. The vector $\vec{\rho}$ characterizes the span of A. We specify the set of weighted parse items via the deduction rules in Fig. 2. Our parser performs a weighted deductive parsing (Nederhof, 2003), based on this deduction system. We use a chart $\mathcal{C}$ and an agenda $\mathcal{A}$, both initially empty, and we proceed as in Fig. 3.

Scan: $\dfrac{}{0 : [A, \langle\langle i, i+1\rangle\rangle]}$ A POS tag of w_{i+1}

Unary: $\dfrac{in : [B, \vec{\rho}]}{in + |log(p)| : [A, \vec{\rho}]}$ $p : A(\vec{\alpha}) \to B(\vec{\alpha}) \in P$

Binary: $\dfrac{in_B : [B, \vec{\rho_B}], in_C : [C, \vec{\rho_C}]}{in_B + in_C + log(p) : [A, \vec{\rho_A}]}$

where $p : A(\vec{\rho_A}) \to B(\vec{\rho_B})C(\vec{\rho_C})$ is an instantiated rule.

Goal: $[S, \langle\langle 0, n\rangle\rangle]$

Figure 2: Weighted CYK deduction system

add SCAN results to $\mathcal{A}$
while $\mathcal{A} \neq \emptyset$
  remove best item $x : I$ from $\mathcal{A}$
  add $x : I$ to $\mathcal{C}$
  if I goal item
  then stop and output true
  else
    for all $y : I'$ deduced from $x : I$ and items in $\mathcal{C}$:
      if there is no z with $z : I' \in \mathcal{C} \cup \mathcal{A}$
      then add $y : I'$ to $\mathcal{A}$
      else if $z : I' \in \mathcal{A}$ for some z
        then update weight of I' in $\mathcal{A}$ to $max(y, z)$

Figure 3: Weighted deductive parsing

4 Outside Estimates

In order to speed up parsing, we add an estimate of the log of the outside probabilities of the items to their weights in the agenda. All our outside estimates are *admissible* (Klein and Manning, 2003a) which means that they never underestimate the actual outside probability of an item. However, most of them are not monotonic. In other words, it can happen that we deduce an item I_2 from an item I_1 where the weight of I_2 is greater than the weight of I_1. The parser can therefore end up in a local maximum that is not the global maximum we are searching for. In other words, our outside weights are only *figures of merit* (FOM). Only for the full SX estimate, the monotonicity is guaranteed and we can do true A* parsing as described in (Klein and Manning, 2003a) that always finds the best parse.

All outside estimates are computed for a certain maximal sentence length len_{max}.

POS tags: $\dfrac{}{0 : [A, \langle 1 \rangle]}$ A a POS tag

Unary: $\dfrac{in : [B, \vec{l}]}{in + log(p) : [A, \vec{l}]}$ $p : A(\vec{\alpha}) \to B(\vec{\alpha}) \in P$

Binary: $\dfrac{in_B : [B, \vec{l_B}], in_C : [C, \vec{l_C}]}{in_B + in_C + log(p) : [A, \vec{l_A}]}$

where $p : A(\vec{\alpha_A}) \to B(\vec{\alpha_B})C(\vec{\alpha_C}) \in P$ and the following holds: we define $\mathcal{B}(i)$ as $\{1 \le j \le dim(B) \,|\, \vec{\alpha_B}(j)$ occurs in $\vec{\alpha_A}(i)\}$ and $\mathcal{C}(i)$ as $\{1 \le j \le dim(C) \,|\, \vec{\alpha_C}(j)$ occurs in $\vec{\alpha_A}(i)\}$. Then for all i, $1 \le i \le dim(A)$: $\vec{l_A}(i) = \Sigma_{j \in \mathcal{B}(i)} \vec{l_B}(j) + \Sigma_{j \in \mathcal{C}(i)} \vec{l_C}(j)$.

Figure 4: Inside estimate

Axiom : $\dfrac{}{0 : [S, \langle 0, len, 0 \rangle]}$ $1 \le len \le len_{max}$

Unary: $\dfrac{w : [A, \vec{l}]}{w + log(p) : [B, \vec{l}]}$ $p : A(\vec{\alpha}) \to B(\vec{\alpha}) \in P$

Binary-right:

$$\frac{w : [X, \vec{l_X}]}{w + in(A, \vec{l'_A}) + log(p) : [B, \vec{l_B}]}$$

Binary-left:

$$\frac{w : [X, \vec{l_X}]}{w + in(B, \vec{l'_B}) + log(p) : [A, \vec{l_A}]}$$

where, for both rules, there is an instantiated rule $p : X(\vec{\rho}) \to A(\vec{\rho_A})B(\vec{\rho_B})$ such that $\vec{l_X} = l_{out}(\rho)$, $\vec{l_A} = l_{out}(\rho_A), \vec{l'_A} = l_{in}(\rho_A), \vec{l_B} = l_{out}(\rho_B, \vec{l_B} = l_{in}(\rho_B$.

Figure 5: Full SX estimate top-down

4.1 Full SX estimate

The full SX estimate, for a given sentence length n, is supposed to give the minimal costs (maximal probability) of completing a category X with a span ρ into an S with span $\langle\langle 0, n\rangle\rangle$.

For the computation, we need an estimate of the inside probability of a category C with a span ρ, regardless of the actual terminals in our input. This inside estimate is computed as shown in Fig. 4. Here, we do not need to consider the number of terminals outside the span of C (to the left or right or in the gaps), they are not relevant for the inside probability. Therefore the items have the form $[A, \langle l_1, \dots, l_{dim(A)}\rangle]$, where A is a non-terminal and l_i gives the length of its ith component. It holds that $\Sigma_{1 \le i \le dim(A)} l_i \le len_{max} - dim(A) + 1$.

A straight-forward extension of the CFG algorithm from (Klein and Manning, 2003a) for computing the SX estimate is given in Fig. 5. For a given range vector $\rho = \langle\langle l_1, r_1\rangle, \dots, \langle l_k, r_k\rangle\rangle$ and a sentence length n, we define its *inside length vector* $l_{in}(\rho)$ as $\langle r_1 - l_1, \dots, r_k - l_k\rangle$ and its *outside length vector* $l_{out}(\rho)$ as $\langle l_1, r_1 - l_1, l_2 - r_1, \dots, l_k - r_{k-1}, r_k - l_k, n - r_k\rangle$.

This algorithm has two major problems: Since it proceeds top-down, in the *Binary* rules, we must compute all splits of the antecedent X span into the spans of A and B which is very expensive. Furthermore, for a category A with a certain number of terminals in the components and the gaps, we compute the lower part of the outside estimate several times, namely for every combination of number of terminals to the left and to the right (first and last element in the outside length vector). In order to avoid these problems, we now abstract away from the lengths of the part to the left and the right, modifying our items such as to allow a bottom-up strategy.

The idea is to compute the weights of items representing the derivations from a certain lower C up to some A (C is a kind of "gap" in the yield of A) while summing up the inside costs of off-spine nodes and the log of the probabilities of the corresponding rules. We use items $[A, C, \rho_A, \rho_C, shift]$ where $A, C \in N$ and ρ_A, ρ_C are range vectors, both with a first component starting at position 0. The integer $shift \le len_{max}$ tells us how many positions to the right the C span is shifted, compared to the starting position of the A. ρ_A and ρ_C represent the spans of C and A while disregarding the number of terminals to the left the right. I.e., only the lengths of the components and of the gaps are encoded. This means in particular that the length n of the sentence does not play a role here. The right boundary of the last range in the vectors is limited to len_{max}. For any $i, 0 \le i \le len_{max}$, and any range vector ρ, we define $shift(\rho, i)$ as the range vector one obtains from adding i to all range boundaries in ρ and $shift(\rho, -i)$ as the range vector one obtains from subtracting i from all boundaries in ρ.

The weight of $[A, C, \rho_A, \rho_C, i]$ estimates the costs for completing a C tree with yield ρ_C into an A tree with yield ρ_A such that, if the span of A starts at position j, the span of C starts at position $i + j$. Fig. 6 gives the computation. The value of $in(A, \vec{l})$ is the inside estimate of $[A, \vec{l}]$.

The SX-estimate for some predicate C with

POS tags: $\dfrac{}{0 : [C, C, \langle 0,1\rangle, \langle 0,1\rangle, 0]}$ C a POS tag

Unary: $\dfrac{0 : [B, B, \rho_B, \rho_B, 0]}{log(p) : [A, B, \rho_B, \rho_B, 0]}$ $p : A(\vec{\alpha}) \to B(\vec{\alpha}) \in P$

Binary-right:
$\dfrac{0 : [A, A, \rho_A, \rho_A, 0], 0 : [B, B, \rho_B, \rho_B, 0]}{in(A, l(\rho_A)) + log(p) : [X, B, \rho_X, \rho_B, i]}$

Binary-left:
$\dfrac{0 : [A, A, \rho_A, \rho_A, 0], 0 : [B, B, \rho_B, \rho_B, 0]}{in(B, l(\rho_B)) + log(p) : [X, A, \rho_X, \rho_A, 0]}$

where i is such that for $shift(\rho_B, i) = \rho'_B$ $p : X(\rho_X) \to A(\rho_A)B(\rho'_B)$ is an instantiated rule.

Starting sub-trees with larger gaps:
$\dfrac{w : [B, C, \rho_B, \rho_C, i]}{0 : [B, B, \rho_B, \rho_B, 0]}$

Transitive closure of sub-tree combination:
$\dfrac{w_1 : [A, B, \rho_A, \rho_B, i], w_2 : [B, C, \rho_B, \rho_C, j]}{w_1 + w_2 : [A, C, \rho_A, \rho_C, i + j]}$

Figure 6: Full SX estimate bottom-up

span ρ where i is the left boundary of the first component of ρ and with sentence length n is then given by the maximal weight of $[S, C, \langle 0, n\rangle, shift(-i, \rho), i]$. Among our estimates, the full SX estimate is the only one that is monotonic and that allows for true A* parsing.

4.2 SX with Left, Gaps, Right, Length

A problem of the previous estimate is that with a large number of non-terminals the computation of the estimate requires too much space. Our experiments have shown that for treebank parsing where we have, after binarization and markovization, appr. 12,000 non-terminals, its computation is not feasible. We therefore turn to simpler estimates with only a single non-terminal per item. We now estimate the outside probability of a non-terminal A with a span of a length *length* (the sum of the lengths of all the components of the span), with *left* terminals to the left of the first component, *right* terminals to the right of the last component and *gaps* terminals in between the components of the A span, i.e., filling the gaps. Our items have the form $[X, len, left, right, gaps]$ with $X \in N$, $len + left + right + gaps \leq len_{max}$, $len \geq dim(X)$, $gaps \geq dim(X) - 1$.

Let us assume that, in the rule $X(\vec{\alpha}) \to A(\vec{\alpha_A})B(\vec{\alpha_B})$, when looking at the vector $\vec{\alpha}$, we have $left_A$ variables for A-components preceding the first variable of a B component, $right_A$ variables for A-components following the last variable of a B component and $right_B$ variables for B-components following the last variable of a A component. (In our grammars, the first LHS argument always starts with the first variable from A.) Furthermore, $gaps_A = dim(A) - left_A - right_A$, $gaps_B = dim(B) - right_B$.

Axiom : $\dfrac{}{0 : [S, len, 0, 0, 0]}$ $1 \leq len \leq len_{max}$

Unary: $\dfrac{w : [X, len, l, r, g]}{w + log(p) : [A, len, l, r, g]}$

where $p : X(\vec{\alpha}) \to A(\vec{\alpha}) \in P$.

Binary-right:
$$\dfrac{w : [X, len, l, r, g]}{w + in(A, len - len_B) + log(p) : [B, len_B, l_B, r_B, g_B]}$$

Binary-left:
$$\dfrac{w : [X, len, l, r, g]}{w + in(B, len - len_A) + log(p) : [A, len_A, l_A, r_A, g_A]}$$

where, for both rules, $p : X(\vec{\alpha}) \to A(\vec{\alpha_A})B(\vec{\alpha_B}) \in P$.

Figure 7: SX with length, left, right, gaps

POS tags: $\dfrac{}{0 : [A, 1]}$ A a POS tag

Unary: $\dfrac{in : [B, l]}{in + log(p) : [A, l]}$ $p : A(\vec{\alpha}) \to B(\vec{\alpha}) \in P$

Binary: $\dfrac{in_B : [B, l_B], in_C : [C, l_C]}{in_B + in_C + log(p) : [A, l_B + l_C]}$

where either $p : A(\vec{\alpha_A}) \to B(\vec{\alpha_B})C(\vec{\alpha_C}) \in P$ or $p : A(\vec{\alpha_A}) \to C(\vec{\alpha_C})B(\vec{\alpha_B}) \in P$.

Figure 8: Inside estimate with total span length

Fig. 7 gives the computation of the estimate. The following side conditions must hold: For *Binary-right* to apply, the following constraints must be satisfied: a) $len + l + r + g = len_B + l_B + r_B + g_B$, b) $l_B \geq l + left_A$, c) if $right_A > 0$, then $r_B \geq r + right_A$, else ($right_A = 0$), $r_B = r$, d) $g_B \geq gaps_A$. Similarly, for *Binary-left* to apply, the following constraints must be satisfied: a) $len + l + r + g = len_A + l_A + r_A + g_A$, b) $l_A = l$, c) if $right_B > 0$, then $r_A \geq r + right_B$, else ($right_B = 0$), $r_A = r$ d) $g_A \geq gaps_B$.

The value $in(X, l)$ for a non-terminal X and a length l, $0 \leq l \leq len_{max}$ is an estimate of the probability of an X category with a span of length l. Its computation is specified in Fig. 8.

The SX-estimate for a sentence length n and for some predicate C with a range characterized by $\vec{\rho} = \langle\langle l_1, r_1\rangle, \dots, \langle l_{dim(C)}, r_{dim(C)}\rangle\rangle$ where $len = \Sigma_{i=1}^{dim(C)}(r_i - l_i)$ and $r = n - r_{dim(C)}$ is then given by the maximal weight of the item $[C, len, l_1, r, n - len - l_1 - r]$.

$$\text{Axiom}: \frac{}{0 : [S, len, 0, 0]}\ 1 \leq len \leq len_{max}$$

$$\text{Unary}: \frac{w : [X, len, lr, g]}{w + log(p) : [A, len, lr, g]}$$

where $p : X(\vec{\alpha}) \rightarrow A(\vec{\alpha}) \in P$.

Binary-right:

$$\frac{w : [X, len, lr, g]}{w + in(A, len - len_B) + log(p) : [B, len_B, lr_B, g_B]}$$

Binary-left:

$$\frac{w : [X, len, lr, g]}{w + in(B, len - len_A) + log(p) : [A, len_A, lr_A, g_A]}$$

where, for both rules, $p : X(\vec{\alpha}) \rightarrow A(\vec{\alpha_A})B(\vec{\alpha_B}) \in P$.

Figure 9: SX estimate with length, LR, gaps

4.3 SX with LR, Gaps, Length

In order to further decrease the space complexity, we can simplify the previous estimate by subsuming the two lengths $left$ and $right$ in a single length lr. I.e., the items now have the form $[X, len, lr, gaps]$ with $X \in N$, $len + lr + gaps \leq len_{max}$, $len \geq dim(X)$, $gaps \geq dim(X) - 1$.

The computation is given in Fig. 9. Again, we define $left_A$, $gaps_A$, $right_A$ and $gaps_B$, $right_B$ for a rule $X(\vec{\alpha}) \rightarrow A(\vec{\alpha_A})B(\vec{\alpha_B})$ as above. The side conditions are as follows: For *Binary-right* to apply, the following constraints must be satisfied: a) $len + lr + g = len_B + lr_B + g_B$, b) $lr < lr_B$, and c) $g_B \geq gaps_A$. For *Binary-left* to apply, the following must hold: a) $len + lr + g = len_A + lr_A + g_A$, b) if $right_B = 0$ then $lr = lr_A$, else $lr < lr_A$ and c) $g_A \geq gaps_B$.

The SX-estimate for a sentence length n and for some predicate C with a span $\vec{\rho} = \langle\langle l_1, r_1\rangle, \ldots, \langle l_{dim(C)}, r_{dim(C)}\rangle\rangle$ where $len = \Sigma_{i=1}^{dim(C)}(r_i - l_i)$ and $r = n - r_{dim(C)}$ is then the maximal weight of $[C, len, l_1 + r, n - len - l_1 - r]$.

5 Evaluation

The goal of our evaluation of our parser is to show that, firstly, reasonable parser speed can be achieved and, secondly, the parser output is of promising quality.

5.1 Data

Our data source is the German NeGra treebank (Skut et al., 1997). In a preprocessing step, following common practice (Kübler and Penn, 2008), we attach punctuation (not included in the NeGra annotation) as follows: In a first pass, using heuristics, we attach punctuation as high as possible while avoiding to introduce new crossing branches. In a second pass, parentheses and quotation marks preferably attach to the same node. Grammatical function labels on the edges are discarded.

We create data sets of different sizes in order to see how the size of the training set relates to the gain using context summary estimates and to the output quality of the parser. The first set uses the first 4000 sentences and the second one all sentences of NeGra. Due to memory limitations, in both sets, we limit ourselves to sentences of a maximal length of 25 words. We use the first 90% of both sets as training set and the remaining 10% as test set. Tab. 1 shows the resulting sizes.

	NeGra-small		NeGra	
	training	test	training	test
size	2839	316	14858	1651

Table 1: Test and training sets

5.2 Treebank Grammar Extraction

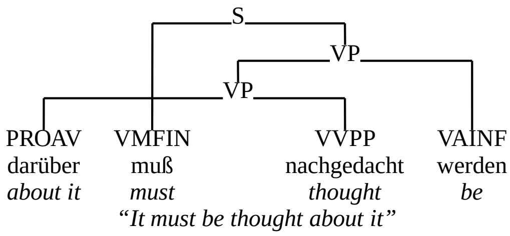


Figure 10: A sample tree from NeGra

As already mentioned, in NeGra, discontinuous phrases are annotated with crossing branches (see Fig. 10 for an example with two discontinuous VPs). Such discontinuities can be straightforwardly modelled with LCFRS. We use the algorithm from Maier and Søgaard (2008) to extract LCFRS rules from NeGra and TIGER. It first creates rules of the form $P(a) \rightarrow \varepsilon$ for each preterminal P dominating some terminal a. Then for all other nonterminals A_0 with the children $A_1 \cdots A_m$, a clause $A_0 \rightarrow A_1 \cdots A_m$ is created. The arguments of the $A_1 \cdots A_m$ are single variables where the number of arguments is the number of discontinuous parts in the yield of a predicate. The arguments of A_0 are concatenations of these variables that describe how the discontinuous parts of the yield of A_0 are obtained from the yields of its daughters. Different occurrences of the same non-terminal, only with different fan-outs, are distinguished by corresponding subscripts. Note that this extraction algorithm yields only *monotone* LCFRS (equivalent to ordered simple RCG). See Maier and Søgaard (2008) for further details. For Fig. 10, we obtain for instance the rules in Fig. 11.

PROAV(Darüber) $\rightarrow \varepsilon$ VMFIN(muß) $\rightarrow \varepsilon$
VVPP(nachgedacht) $\rightarrow \varepsilon$ VAINF(werden) $\rightarrow \varepsilon$
$\mathrm{S}_1(X_1X_2X_3) \rightarrow \mathrm{VP}_2(X_1, X_3)\ \mathrm{VMFIN}(X_2)$
$\mathrm{VP}_2(X_1, X_2X_3) \rightarrow \mathrm{VP}_2(X_1, X_2)\ \mathrm{VAINF}(X_3)$
$\mathrm{VP}_2(X_1, X_2) \rightarrow \mathrm{PROAV}(X_1)\ \mathrm{VVPP}(X_2)$

Figure 11: LCFRS rules for the tree in Fig. 10

5.3 Binarization and Markovization

Before parsing, we binarize the extracted LCFRS. For this we first apply Collins-style head rules, based on the rules the Stanford parser (Klein and Manning, 2003b) uses for NeGra, to mark the resp. head daughters of all non-terminal nodes. Then, we reorder the RHSs such that the sequence γ of elements to the right of the head daughter is reversed and moved to the beginning of the RHS. We then perform a binarization that proceeds from left to right. The binarization works like the transformation into Chomsky Normal Form for CFGs in the sense that for RHSs longer than 2, we introduce a new non-terminal that covers the RHS without the first element. The rightmost new rule, which covers the head daughter, is binarized to unary. We do not use a unique new non-terminal for every new rule. Instead, to the new symbols introduced during the binarization (VP_{bin} in the example), a variable number of symbols from the vertical and horizontal context of the original rule is added in order to achieve *markovization*. Following the literature, we call the respective quantities v and h. For reasons of space we restrict ourselves here to the example in Fig. 12. Refer to Maier and Kallmeyer (2010) for a detailed presentation of the binarization and markovization.

The probabilities are then computed based on the rule frequencies in the transformed treebank, using a Maximum Likelihood estimator.

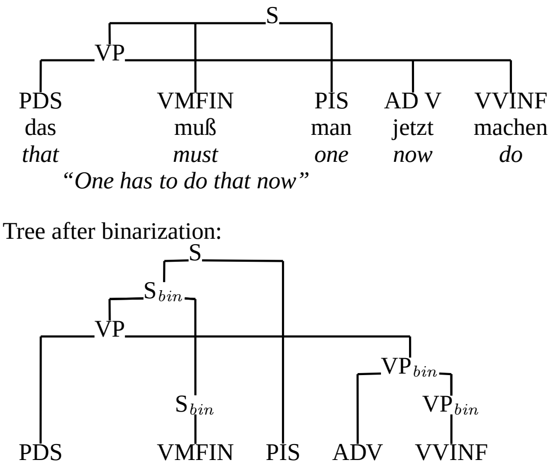


Figure 12: Sample binarization

5.4 Evaluation of Parsing Results

In order to assess the quality of the output of our parser, we choose an EVALB-style metric, i.e., we compare phrase boundaries. In the context of LCFRS, we compare sets of items $[A, \vec{\rho}]$ that characterize the span of a non-terminal A in a derivation tree. One set is obtained from the parser output, and one from the corresponding treebank trees. Using these item sets, we compute labeled and unlabeled recall (LR/UR), precision (LP/UP), and the F_1 measure (LF_1/UF_1). Note that if $k = 1$, our metric is identical to its PCFG equivalent.We are aware of the recent discussion about the shortcomings of EVALB. A discussion of this issue is presented in Maier (2010).

5.5 Experiments

In all experiments, we provide the parser with gold part-of-speech tags. For the experiments with *NeGra-small*, the parser is given the markovization settings $v = 1$ and $h = 1$. We compare the parser performance without estimates (OFF) with its performance with the estimates described in 4.2 (SIMPLE) and 4.3 (LR). Tab. 2 shows the results. Fig. 13 shows the number of items produced by the parser, indicating that the estimates have the desired effect of preventing unnecessary items from being produced. Note that it is even the case that the parser produces less items for the big set with LR than for the small set without estimate.

We can see that the estimates lead to a slightly

	OFF	SIMPLE	LR
UP/UR	72.29/72.40	70.49/71.81	72.10/72.60
UF_1	72.35	71.14	72.35
LP/LR	68.31/68.41	64.93/66.14	67.35/66.14
LF_1	68.36	65.53	65.53
Parsed	313 (99.05%)	313 (99.05%)	313 (99.05%)

Table 2: Experiments with *NeGra-small*

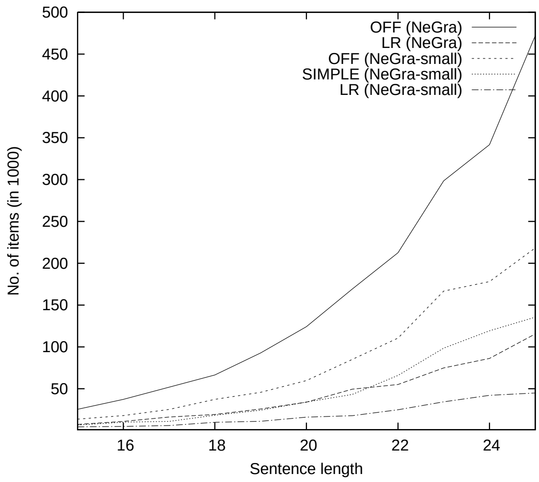


Figure 13: Items produced by the parser

lower F-score. However, while the losses in terms of F_1 are small, the gains in parsing time are substantial, as Fig. 13 shows.

Tab. 3 shows the results of experiments with *NeGra*, with the markovization settings $v = 2$ and $h = 1$ which have proven to be successful for PCFG parsing of NeGra (Rafferty and Manning, 2008). Unfortunately, due to memory restrictions, we were not able to compute SIMPLE for the large data set.[1] Resp. LR, the findings are comparable to the ones for *NeGra-short*. The speedup is paid with a lower F_1.

	OFF	LR
UP/UR	76.89/77.35	75.22/75.99
UF_1	77.12	75.60
LP/LR	73.03/73.46	70.98/71.70
LF_1	73.25	71.33
Parsed	1642 (99.45%)	1642 (99.45%)

Table 3: Experiments with *NeGra*

Our results are not directly comparable with PCFG parsing results, since LCFRS parsing is a harder task. However, since the EVALB metric coincides for constituents without crossing branches, in order to place our results in the context of previous work on parsing NeGra, we cite some of the results from the literature which were obtained using PCFG parsers[2]: Kübler (2005) (Tab. 1, plain PCFG) obtains 69.4, Dubey and Keller (2003) (Tab. 5, sister-head PCFG model) 71.12, Rafferty and Manning (2008) (Tab. 2, Stanford parser with markovization $v = 2$ and $h = 1$) 77.2, and Petrov and Klein (2007) (Tab. 1, Berkeley parser) 80.1. Plaehn (2004) obtains 73.16 Labeled F_1 using Probabilistic Discontinuous Phrase Structure Grammar (DPSG), albeit only on sentences with a length of up to 15 words. On those sentences, we obtain 81.27.

The comparison shows that our system delivers competitive results. Additionally, when comparing this to PCFG parsing results, one has to keep in mind that LCFRS parse trees contain non-context-free information about discontinuities. Therefore, a correct parse with our grammar is actually better than a correct CFG parse, evaluated with respect to a transformation of NeGra into a context-free treebank where precisely this information gets lost.

6 Conclusion

We have presented the first parser for unrestricted Probabilistic Linear Context-Free Rewriting Systems (PLCFRS), implemented as a CYK parser with weighted deductive parsing. To speed up parsing, we use context summary estimates for parse items. An evaluation on the NeGra treebank, both in terms of output quality and speed, shows that data-driven parsing using PLCFRS is feasible. Already in this first attempt with a straightforward binarization, we obtain results that are comparable to state-of-the-art PCFG results in terms of F_1, while yielding parse trees that are richer than context-free trees since they describe discontinuities. Therefore, our approach demonstrates convincingly that PLCFRS is a natural and tractable alternative for data-driven parsing which takes non-local dependencies into consideration.

[1] SIMPLE also proved to be infeasible to compute for the small set for the markovization settings $v = 2$ and $h = 1$ due to the greatly increased label set with this settings.

[2] Note that these results were obtained on sentences with a length of ≤ 40 words and that those parser possibly would deliver better results if tested on our test set.

References

Boullier, Pierre. 1998. A Proposal for a Natural Language Processing Syntactic Backbone. Technical Report 3342, INRIA.

Boyd, Adriane. 2007. Discontinuity revisited: An improved conversion to context-free representations. In *The Linguistic Annotation Workshop at ACL 2007*.

Charniak, Eugene and Sharon A. Caraballo. 1998. New figures of merit for best-first probabilistic chart parsing. *Computational Linguistics*, 24.

Collins, Michael. 1999. *Head-driven statistical models for natural language parsing*. Ph.D. thesis, University of Pennsylvania.

Dubey, Amit and Frank Keller. 2003. Probabilistic parsing for German using sisterhead dependencies. In *Proceedings of ACL*.

Gómez-Rodríguez, Carlos, Marco Kuhlmann, Giorgio Satta, and David Weir. 2009. Optimal reduction of rule length in linear context-free rewriting systems. In *Proceedings of NAACL-HLT*.

Hockenmaier, Julia. 2003. *Data and models for Statistical Parsing with Combinatory Categorial Grammar*. Ph.D. thesis, University of Edinburgh.

Johnson, Mark. 2002. A simple pattern-matching algorithm for recovering empty nodes and their antecedents. In *Proceedings of ACL*.

Kato, Yuki, Hiroyuki Seki, and Tadao Kasami. 2006. Stochastic multiple context-free grammar for rna pseudoknot modeling. In *Proceedings of TAG+8*.

Klein, Dan and Christopher D. Manning. 2003a. A* Parsing: Fast Exact Viterbi Parse Selection. In *Proceedings of NAACL-HLT*.

Klein, Dan and Christopher D. Manning. 2003b. Fast exact inference with a factored model for natural language parsing. In *In Advances in Neural Information Processing Systems 15 (NIPS)*.

Kübler, Sandra and Gerald Penn, editors. 2008. *Proceedings of the Workshop on Parsing German at ACL 2008*.

Kübler, Sandra. 2005. How do treebank annotation schemes influence parsing results? Or how not to compare apples and oranges. In *Proceedings of RANLP 2005*.

Kuhlmann, Marco and Giorgio Satta. 2009. Treebank grammar techniques for non-projective dependency parsing. In *Proceedings of EACL*.

Levy, Roger and Christopher D. Manning. 2004. Deep dependencies from context-free statistical parsers: correcting the surface dependency approximation. In *Proceedings of ACL*.

Maier, Wolfgang and Laura Kallmeyer. 2010. Discontinuity and non-projectivity: Using mildly context-sensitive formalisms for data-driven parsing. In *Proceedings of TAG+10*.

Maier, Wolfgang and Timm Lichte. 2009. Characterizing Discontinuity in Constituent Treebanks. In *Proceedings of Formal Grammar 2009*.

Maier, Wolfgang and Anders Søgaard. 2008. Treebanks and mild context-sensitivity. In *Proceedings of Formal Grammar 2008*.

Maier, Wolfgang. 2010. Direct parsing of discontinuous constituents in german. In *Proceedings of the SPMRL workshop at NAACL HLT 2010*.

Marcus, Mitchell, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of HLT*.

Nederhof, Mark-Jan. 2003. Weighted Deductive Parsing and Knuth's Algorithm. *Computational Linguistics*, 29(1).

Petrov, Slav and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HLT-NAACL 2007*.

Plaehn, Oliver. 2004. Computing the most probable parse for a discontinuous phrase-structure grammar. In *New developments in parsing technology*. Kluwer.

Rafferty, Anna and Christopher D. Manning, 2008. *Parsing Three German Treebanks: Lexicalized and Unlexicalized Baselines*. In Kübler and Penn (2008).

Seki, Hiroyuki, Takahashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88(2).

Skut, Wojciech, Brigitte Krenn, Thorten Brants, and Hans Uszkoreit. 1997. An Annotation Scheme for Free Word Order Languages. In *Proceedings of ANLP*.

Vijay-Shanker, K., David J. Weir, and Aravind K. Joshi. 1987. Characterizing structural descriptions produced by various grammatical formalisms. In *Proceedings of ACL*.

Learning to Predict Readability using Diverse Linguistic Features

Rohit J. Kate[1] **Xiaoqiang Luo**[2] **Siddharth Patwardhan**[2] **Martin Franz**[2]
Radu Florian[2] **Raymond J. Mooney**[1] **Salim Roukos**[2] **Chris Welty**[2]
[1]Department of Computer Science
The University of Texas at Austin
{rjkate,mooney}@cs.utexas.edu
[2]IBM Watson Research Center
{xiaoluo,spatward,franzm,raduf,roukos,welty}@us.ibm.com

Abstract

In this paper we consider the problem of building a system to predict readability of natural-language documents. Our system is trained using diverse features based on syntax and language models which are generally indicative of readability. The experimental results on a dataset of documents from a mix of genres show that the predictions of the learned system are more accurate than the predictions of naive human judges when compared against the predictions of linguistically-trained expert human judges. The experiments also compare the performances of different learning algorithms and different types of feature sets when used for predicting readability.

1 Introduction

An important aspect of a document is whether it is easily processed and understood by a human reader as intended by its writer, this is termed as the document's *readability*. Readability involves many aspects including grammaticality, conciseness, clarity, and lack of ambiguity. Teachers, journalists, editors, and other professionals routinely make judgements on the readability of documents. We explore the task of learning to automatically judge the readability of natural-language documents.

In a variety of applications it would be useful to be able to automate readability judgements. For example, the results of a web-search can be ordered taking into account the readability of the retrieved documents thus improving user satisfaction. Readability judgements can also be used for automatically grading essays, selecting instructional reading materials, etc. If documents are generated by machines, such as summarization or machine translation systems, then they are prone to be less readable. In such cases, a readability measure can be used to automatically filter out documents which have poor readability. Even when the intended consumers of text are machines, for example, information extraction or knowledge extraction systems, a readability measure can be used to filter out documents of poor readability so that the machine readers will not extract incorrect information because of ambiguity or lack of clarity in the documents.

As part of the DARPA Machine Reading Program (MRP), an evaluation was designed and conducted for the task of rating documents for readability. In this evaluation, 540 documents were rated for readability by both experts and novice human subjects. Systems were evaluated based on whether they were able to match expert readability ratings better than novice raters. Our system learns to match expert readability ratings by employing regression over a set of diverse linguistic features that were deemed potentially relevant to readability. Our results demonstrate that a rich combination of features from syntactic parsers, language models, as well as lexical statistics all contribute to accurately predicting expert human readability judgements. We have also considered the effect of different genres in predicting readability and how the genre-specific language models can be exploited to improve the readability predictions.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 546–554,
Beijing, August 2010

2 Related Work

There is a significant amount of published work on a related problem: predicting the reading difficulty of documents, typically, as the school grade-level of the reader from grade 1 to 12. Some early methods measure simple characteristics of documents like average sentence length, average number of syllables per word, etc. and combine them using a linear formula to predict the grade level of a document, for example FOG (Gunning, 1952), SMOG (McLaughlin, 1969) and Flesh-Kincaid (Kincaid et al., 1975) metrics. These methods do not take into account the content of the documents. Some later methods use pre-determined lists of words to determine the grade level of a document, for example the Lexile measure (Stenner et al., 1988), the Fry Short Passage measure (Fry, 1990) and the Revised Dale-Chall formula (Chall and Dale, 1995). The word lists these methods use may be thought of as very simple language models. More recently, language models have been used for predicting the grade level of documents. Si and Callan (2001) and Collins-Thompson and Callan (2004) train unigram language models to predict grade levels of documents. In addition to language models, Heilman et al. (2007) and Schwarm and Ostendorf (2005) also use some syntactic features to estimate the grade level of texts.

Pitler and Nenkova (2008) consider a different task of predicting text quality for an educated adult audience. Their system predicts readability of texts from Wall Street Journal using lexical, syntactic and discourse features. Kanungo and Orr (2009) consider the task of predicting readability of web summary snippets produced by search engines. Using simple surface level features like the number of characters and syllables per word, capitalization, punctuation, ellipses etc. they train a regression model to predict readability values.

Our work differs from this previous research in several ways. Firstly, the task we have considered is different, we predict the readability of general documents, not their grade level. The documents in our data are also not from any single domain, genre or reader group, which makes our task more general. The data includes human written as well as machine generated documents. The task and the data has been set this way because it is aimed at filtering out documents of poor quality for later processing, like for extracting machine-processable knowledge from them. Extracting knowledge from openly found text, such as from the internet, is becoming popular but the quality of text found "in the wild", like found through searching the internet, vary considerably in quality and genre. If the text is of poor readability then it is likely to lead to extraction errors and more problems downstream. If the readers are going to be humans instead of machines, then also it is best to filter out poorly written documents. Hence identifying readability of general text documents coming from various sources and genres is an important task. We are not aware of any other work which has considered such a task.

Secondly, we note that all of the above approaches that use language models train a language model for each difficulty level using the training data for that level. However, since the amount of training data annotated with levels is limited, they can not train higher-order language models, and most just use unigram models. In contrast, we employ more powerful language models trained on large quantities of generic text (which is not from the training data for readability) and use various features obtained from these language models to predict readability. Thirdly, we use a more sophisticated combination of linguistic features derived from various syntactic parsers and language models than any previous work. We also present ablation results for different sets of features. Fourthly, given that the documents in our data are not from a particular genre but from a mix of genres, we also train genre-specific language models and show that including these as features improves readability predictions. Finally, we also show comparison between various machine learning algorithms for predicting readability, none of the previous work compared learning algorithms.

3 Readability Data

The readability data was collected and released by LDC. The documents were collected from the following diverse sources or genres: newswire/newspaper text, weblogs, newsgroup posts, manual transcripts, machine translation output, closed-caption transcripts and Wikipedia articles. Documents for newswire, machine translation and closed captioned genres were collected automatically by first forming a candidate pool from a single collection stream and then randomly selecting documents. Documents for weblogs, newsgroups and manual transcripts were also collected in the same way but were then reviewed by humans to make sure they were not simply spam articles or something objectionable. The Wikipedia articles were collected manually, by searching through a data archive or the live web, using keyword and other search techniques. Note that the information about genres of the documents is not available during testing and hence was not used when training our readability model.

A total of 540 documents were collected in this way which were uniformly distributed across the seven genres. Each document was then judged for its readability by eight expert human judges. These expert judges are native English speakers who are language professionals and who have specialized training in linguistic analysis and annotation, including the machine translation post-editing task. Each document was also judged for its readability by six to ten naive human judges. These non-expert (naive) judges are native English speakers who are not language professionals (e.g. editors, writers, English teachers, linguistic annotators, etc.) and have no specialized language analysis or linguistic annotation training. Both expert and naive judges provided readability judgments using a customized web interface and gave a rating on a 5-point scale to indicate how readable the passage is (where 1 is lowest and 5 is highest readability) where readability is defined as a subjective judgment of how easily a reader can extract the information the writer or speaker intended to convey.

4 Readability Model

We want to answer the question whether a machine can accurately estimate readability as judged by a human. Therefore, we built a machine-learning system that predicts the readability of documents by training on expert human judgements of readability. The evaluation was then designed to compare how well machine and naive human judges predict expert human judgements. In order to make the machine's predicted score comparable to a human judge's score (details about our evaluation metrics are in Section 6.1), we also restricted the machine scores to integers. Hence, the task is to predict an integer score from 1 to 5 that measures the readability of the document.

This task could be modeled as a multi-class classification problem treating each integer score as a separate class, as done in some of the previous work (Si and Callan, 2001; Collins-Thompson and Callan, 2004). However, since the classes are numerical and not unrelated (for example, the score 2 is in between scores 1 and 3), we decided to model the task as a regression problem and then round the predicted score to obtain the closest integer value. Preliminary results verified that regression performed better than classification. Heilman et al. (2008) also found that it is better to treat the readability scores as ordinal than as nominal. We take the average of the expert judge scores for each document as its gold-standard score. Regression was also used by Kanungo and Orr (2009), although their evaluation did not constrain machine scores to be integers.

We tested several regression algorithms available in the Weka[1] machine learning package, and in Section 6.2 we report results for several which performed best. The next section describes the numerically-valued features that we used as input for regression.

5 Features for Predicting Readability

Good input features are critical to the success of any regression algorithm. We used three main categories of features to predict readability: syntactic features, language-model features, and lexical features, as described below.

5.1 Features Based on Syntax

Many times, a document is found to be unreadable due to unusual linguistic constructs or ungram-

[1] http://www.cs.waikato.ac.nz/ml/weka/

matical language that tend to manifest themselves in the syntactic properties of the text. Therefore, syntactic features have been previously used (Bernth, 1997) to gauge the "clarity" of written text, with the goal of helping writers improve their writing skills. Here too, we use several features based on syntactic analyses. Syntactic analyses are obtained from the Sundance shallow parser (Riloff and Phillips, 2004) and from the English Slot Grammar (ESG) (McCord, 1989).

Sundance features: The Sundance system is a rule-based system that performs a shallow syntactic analysis of text. We expect that this analysis over readable text would be "well-formed", adhering to grammatical rules of the English language. Deviations from these rules can be indications of unreadable text. We attempt to capture such deviations from grammatical rules through the following Sundance features computed for each text document: proportion of sentences with no verb phrases, average number of clauses per sentence, average sentence length in tokens, average number of noun phrases per sentence, average number of verb phrases per sentence, average number of prepositional phrases per sentence, average number of phrases (all types) per sentence and average number of phrases (all types) per clause.

ESG features: ESG uses slot grammar rules to perform a deeper linguistic analysis of sentences than the Sundance system. ESG may consider several different interpretations of a sentence, before deciding to choose one over the other interpretations. Sometimes ESG's grammar rules fail to produce a single complete interpretation of a sentence, in which case it generates partial parses. This typically happens in cases when sentences are ungrammatical, and possibly, less readable. Thus, we use the proportion of such incomplete parses within a document as a readability feature. In case of extremely short documents, this proportion of incomplete parses can be misleading. To account for such short documents, we introduce a variation of the above incomplete parse feature, by weighting it with a log factor as was done in (Riloff, 1996; Thelen and Riloff, 2002).

We also experimented with some other syntactic features such as average sentence parse scores from Stanford parser and an in-house maximum entropy statistical parer, average constituent scores etc., however, they slightly degraded the performance in combination with the rest of the features and hence we did not include them in the final set. One possible explanation could be that averaging diminishes the effect of low scores caused by ungrammaticality.

5.2 Features Based on Language Models

A probabilistic language model provides a prediction of how likely a given sentence was generated by the same underlying process that generated a corpus of training documents. In addition to a general n-gram language model trained on a large body of text, we also exploit language models trained to recognize specific "genres" of text. If a document is translated by a machine, or casually produced by humans for a weblog or newsgroup, it exhibits a character that is distinct from documents that go through a dedicated editing process (e.g., newswire and Wikipedia articles). Below we describe features based on generic as well as genre-specific language models.

Normalized document probability: One obvious proxy for readability is the score assigned to a document by a generic language model (LM). Since the language model is trained on well-written English text, it penalizes documents deviating from the statistics collected from the LM training documents. Due to variable document lengths, we normalize the document-level LM score by the number of words and compute the normalized document probability $NP(\mathcal{D})$ for a document $\mathcal{D}$ as follows:

$$NP(\mathcal{D}) = \big(P(\mathcal{D}|\mathcal{M})\big)^{\frac{1}{|\mathcal{D}|}}, \tag{1}$$

where $\mathcal{M}$ is a general-purpose language model trained on clean English text, and $|\mathcal{D}|$ is the number of words in the document $\mathcal{D}$.

Perplexities from genre-specific language models: The usefulness of LM-based features in categorizing text (McCallum and Nigam, 1998; Yang and Liu, 1999) and evaluating readability (Collins-Thompson and Callan, 2004; Heilman et al., 2007) has been investigated in previous work. In our experiments, however, since documents were acquired through several different channels, such as machine translation or web logs, we also build models that try to predict the genre of a document. Since the genre information for many English documents is readily available, we trained a series of genre-specific 5-gram LMs using the modified Kneser-Ney smoothing (Kneser and Ney, 1995; Stanley and Goodman, 1996). Table 1 contains a list of a base LM and genre-specific LMs.

Given a document $\mathcal{D}$ consisting of tokenized word sequence $\{w_i : i = 1, 2, \cdots, |\mathcal{D}|\}$, its perplexity $L(\mathcal{D}|\mathcal{M}_j)$ with respect to a LM $\mathcal{M}_j$ is computed as:

$$L(\mathcal{D}|\mathcal{M}_j) = \quad e^{\left(-\frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}\log P(w_i|h_i;\mathcal{M}_j)\right)}, \quad (2)$$

where $|\mathcal{D}|$ is the number of words in $\mathcal{D}$ and h_i are the history words for w_i, and $P(w_i|h_i;\mathcal{M}_j)$ is the probability $\mathcal{M}_j$ assigns to w_i, when it follows the history words h_i.

Posterior perplexities from genre-specific language models: While perplexities computed from genre-specific LMs reflect the absolute probability that a document was generated by a specific model, a model's *relative* probability compared to other models may be a more useful feature. To this end, we also compute the posterior perplexity defined as follows. Let $\mathcal{D}$ be a document, $\{\mathcal{M}_i\}_{i=1}^{G}$ be G genre-specific LMs, and $L(\mathcal{D}|\mathcal{M}_i)$ be the perplexity of the document $\mathcal{D}$ with respect to $\mathcal{M}_i$, then the posterior perplexity, $R(\mathcal{M}_i|\mathcal{D})$, is defined as:

$$R(\mathcal{M}_i|\mathcal{D}) = \frac{L(\mathcal{D}|\mathcal{M}_i)}{\sum_{j=1}^{G} L(\mathcal{D}|\mathcal{M}_j)}. \quad (3)$$

We use the term "posterior" because if a uniform prior is adopted for $\{\mathcal{M}_i\}_{i=1}^{G}$, $R(\mathcal{M}_i|\mathcal{D})$ can be interpreted as the posterior probability of the genre LM $\mathcal{M}_i$ given the document $\mathcal{D}$.

5.3 Lexical Features

The final set of features involve various lexical statistics as described below.

Out-of-vocabulary (OOV) rates: We conjecture that documents containing typographical errors (e.g., for closed-caption and web log documents) may receive low readability ratings. Therefore, we compute the OOV rates of a document with respect to the various LMs shown in Table 1. Since modern LMs often have a very large vocabulary, to get meaningful OOV rates, we truncate the vocabularies to the top (i.e., most frequent) 3000 words. For the purpose of OOV computation, a document $\mathcal{D}$ is treated as a sequence of tokenized words $\{w_i : i = 1, 2, \cdots, |\mathcal{D}|\}$. Its OOV rate with respect to a (truncated) vocabulary $\mathcal{V}$ is then:

$$OOV(\mathcal{D}|\mathcal{V}) = \frac{\sum_{i=1}^{\mathcal{D}} I(w_i \notin \mathcal{V})}{|\mathcal{D}|}, \quad (4)$$

where $I(w_i \notin \mathcal{V})$ is an indicator function taking value 1 if w_i is not in $\mathcal{V}$, and 0 otherwise.

Ratio of function words: A characteristic of documents generated by foreign speakers and machine translation is a failure to produce certain function words, such as "the," or "of." So we predefine a small set of function words (mainly English articles and frequent prepositions) and compute the ratio of function words over the total number words in a document:

$$RF(\mathcal{D}) = \frac{\sum_{i=1}^{\mathcal{D}} I(w_i \in \mathcal{F})}{|\mathcal{D}|}, \quad (5)$$

where $I(w_i \in \mathcal{F})$ is 1 if w_i is in the set of function words $\mathcal{F}$, and 0 otherwise.

Ratio of pronouns: Many foreign languages that are source languages of machine-translated documents are pronoun-drop languages, such as Arabic, Chinese, and romance languages. We conjecture that the pronoun ratio may be a good indicator whether a document is translated by machine or produced by humans, and for each document, we first run a POS tagger, and then compute the ratio of pronouns over the number of words in the document:

$$RP(\mathcal{D}) = \frac{\sum_{i=1}^{\mathcal{D}} I(POS(w_i) \in \mathcal{P})}{|\mathcal{D}|}, \quad (6)$$

where $I(POS(w_i) \in \mathcal{F})$ is 1 if the POS tag of w_i is in the set of pronouns, $\mathcal{P}$, and 0 otherwise.

Fraction of known words: This feature measures the fraction of words in a document that occur either in an English dictionary or a gazetteer of names of people and locations.

6 Experiments

This section describes the evaluation methodology and metrics and presents and discusses our

Genre	Training Size(M tokens)	Data Sources
base	5136.8	mostly LDC's GigaWord set
NW	143.2	newswire subset of base
NG	218.6	newsgroup subset of base
WL	18.5	weblog subset of base
BC	1.6	broadcast conversation subset of base
BN	1.1	broadcast news subset of base
wikipedia	2264.6	Wikipedia text
CC	0.1	closed caption
ZhEn	79.6	output of Chinese to English Machine Translation
ArEn	126.8	output of Arabic to English Machine Translation

Table 1: Genre-specific LMs: the second column contains the number of tokens in LM training data (in million tokens).

experimental results. The results of the official evaluation task are also reported.

6.1 Evaluation Metric

The evaluation process for the DARPA MRP readability test was designed by the evaluation team led by SAIC. In order to compare a machine's predicted readability score to those assigned by the expert judges, the Pearson correlation coefficient was computed. The mean of the expert-judge scores was taken as the gold-standard score for a document.

To determine whether the machine predicts scores closer to the expert judges' scores than what an average naive judge would predict, a sampling distribution representing the underlying novice performance was computed. This was obtained by choosing a random naive judge for every document, calculating the Pearson correlation coefficient with the expert gold-standard scores and then repeating this procedure a sufficient number of times (5000). The upper critical value was set at 97.5% confidence, meaning that if the machine performs better than the upper critical value then we reject the null hypothesis that machine scores and naive scores come from the same distribution and conclude that the machine performs significantly better than naive judges in matching the expert judges.

6.2 Results and Discussion

We evaluated our readability system on the dataset of 390 documents which was released earlier during the training phase of the evaluation task. We used stratified 13-fold cross-validation in which the documents from various genres in each fold was distributed in roughly the same proportion as in the overall dataset. We first conducted experiments to test different regression algorithms using all the available features. Next, we ablated various feature sets to determine how much each feature set was contributing to making accurate readability judgements. These experiments are described in the following subsections.

Algorithm	Correlation
Bagged Decision Trees	0.8173
Decision Trees	0.7260
Linear Regression	0.7984
SVM Regression	0.7915
Gaussian Process Regression	0.7562
Naive Judges	
Upper Critical Value	0.7015
Distribution Mean	0.6517
Baselines	
Uniform Random	0.0157
Proportional Random	-0.0834

Table 2: Comparing different algorithms on the readability task using 13-fold cross-validation on the 390 documents using all the features. Exceeding the upper critical value of the naive judges' distribution indicates statistically significantly better predictions than the naive judges.

6.2.1 Regression Algorithms

We used several regression algorithms available in the Weka machine learning package and Table 2 shows the results obtained. The default values

Feature Set	Correlation
Lexical	0.5760
Syntactic	0.7010
Lexical + Syntactic	0.7274
Language Model based	0.7864
All	0.8173

Table 3: Comparison of different linguistic feature sets.

in Weka were used for all parameters, changing these values did not show any improvement. We used decision tree (reduced error pruning (Quinlan, 1987)) regression, decision tree regression with bagging (Breiman, 1996), support vector regression (Smola and Scholkopf, 1998) using polynomial kernel of degree two,[2] linear regression and Gaussian process regression (Rasmussen and Williams, 2006). The distribution mean and the upper critical values of the correlation coefficient distribution for the naive judges are also shown in the table.

Since they are above the upper critical value, all algorithms predicted expert readability scores significantly more accurately than the naive judges. Bagged decision trees performed slightly better than other methods. As shown in the following section, ablating features affects predictive accuracy much more than changing the regression algorithm. Therefore, on this task, the choice of regression algorithm was not very critical once good readability features are used. We also tested two simple baseline strategies: predicting a score uniformly at random, and predicting a score proportional to its frequency in the training data. As shown in the last two rows of Table 2, these baselines perform very poorly, verifying that predicting readability on this dataset as evaluated by our evaluation metric is not trivial.

6.2.2 Ablations with Feature Sets

We evaluated the contributions of different feature sets through ablation experiments. Bagged decision-tree was used as the regression algorithm in all of these experiments. First we compared syntactic, lexical and language-model based features as described in Section 5, and Table 3 shows the results. The language-model feature set performs the best, but performance improves when it is combined with the remaining features. The lexical feature set by itself performs the worst, even below the naive distribution mean (shown in Table 2); however, when combined with syntactic features it performs well.

In our second ablation experiment, we compared the performance of genre-independent and genre-based features. Since the genre-based features exploit knowledge of the genres of text used in the MRP readability corpus, their utility is somewhat tailored to this specific corpus. Therefore, it is useful to evaluate the performance of the system when genre information is not exploited. Of the lexical features described in subsection 5.3, the ratio of function words, ratio of pronoun words and all of the out-of-vocabulary rates except for the base language model are genre-based features. Out of the language model features described in the Subsection 5.2, all of the perplexities except for the base language model and all of the posterior perplexities[3] are genre-based features. All of the remaining features are genre-independent. Table 4 shows the results comparing these two feature sets. The genre-based features do well by themselves but the rest of the features help further improve the performance. While the genre-independent features by themselves do not exceed the upper critical value of the naive judges' distribution, they are very close to it and still outperform its mean value. These results show that for a dataset like ours, which is composed of a mix of genres that themselves are indicative of readability, features that help identify the genre of a text improve performance significantly.[4] For applications mentioned in the introduction and related work sections, such as filtering less readable documents from web-search, many of the input documents could come from some of the common genres considered in our dataset.

In our final ablation experiment, we evaluated

[2]Polynomial kernels with other degrees and RBF kernel performed worse.

[3]Base model for posterior perplexities is computed using other genre-based LMs (equation 3) hence it can not be considered genre-independent.

[4]We note that none of the genre-based features were trained on supervised readability data, but were trained on readily-available large unannotated corpora as shown in Table 1.

Feature Set	Correlation
Genre-independent	0.6978
Genre-based	0.7749
All	0.8173

Table 4: Comparison of genre-independent and genre-based feature sets.

Feature Set	By itself	Ablated from All
Sundance features	0.5417	0.7993
ESG features	0.5841	0.8118
Perplexities	0.7092	0.8081
Posterior perplexities	0.7832	0.7439
Out-of-vocabulary rates	0.3574	0.8125
All	0.8173	-

Table 5: Ablations with some individual feature sets.

the contribution of various individual feature sets. Table 5 shows that posterior perplexities perform the strongest on their own, but without them, the remaining features also do well. When used by themselves, some feature sets perform below the naive judges' distribution mean, however, removing them from the rest of the feature sets degrades the performance. This shows that no individual feature set is critical for good performance but each further improves the performance when added to the rest of the feature sets.

6.3 Official Evaluation Results

An official evaluation was conducted by the evaluation team SAIC on behalf of DARPA in which three teams participated including ours. The evaluation task required predicting the readability of 150 test documents using the 390 training documents. Besides the correlation metric, two additional metrics were used. One of them computed for a document the difference between the average absolute difference of the naive judge scores from the mean expert score and the absolute difference of the machine's score from the mean expert score. This was then averaged over all the documents. The other one was "target hits" which measured if the predicted score for a document fell within the width of the lowest and the highest expert scores for that document, and if so, computed a score inversely proportional to that width. The final target hits score was then computed by averaging it across all the documents. The upper critical values for these metrics were computed in a way analogous to that for the correlation metric which was described before. Higher score is better for all the three metrics. Table 6 shows the results of the evaluation. Our system performed favorably and always scored better than the upper critical value on each of the metrics. Its performance was in between the performance of the other two systems. The performances of the systems show that the correlation metric was the most difficult of the three metrics.

System	Correl.	Avg. Diff.	Target Hits
Our (A)	0.8127	0.4844	0.4619
System B	0.6904	0.3916	0.4530
System C	0.8501	0.5177	0.4641
Upper CV	0.7423	0.0960	0.3713

Table 6: Results of the systems that participated in the DARPA's readability evaluation task. The three metrics used were correlation, average absolute difference and target hits measured against the expert readability scores. The upper critical values are for the score distributions of naive judges.

7 Conclusions

Using regression over a diverse combination of syntactic, lexical and language-model based features, we built a system for predicting the readability of natural-language documents. The system accurately predicts readability as judged by linguistically-trained expert human judges and exceeds the accuracy of naive human judges. Language-model based features were found to be most useful for this task, but syntactic and lexical features were also helpful. We also found that for a corpus consisting of documents from a diverse mix of genres, using features that are indicative of the genre significantly improve the accuracy of readability predictions. Such a system could be used to filter out less readable documents for machine or human processing.

Acknowledgment

This research was funded by Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program.

References

Bernth, Arendse. 1997. Easyenglish: A tool for improving document quality. In *Proceedings of the fifth conference on Applied Natural Language Processing*, pages 159–165, Washington DC, April.

Breiman, Leo. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.

Chall, J.S. and E. Dale. 1995. *Readability Revisited: The New Dale-Chall Readability Formula*. Brookline Books, Cambridge, MA.

Collins-Thompson, Kevyn and James P. Callan. 2004. A language modeling approach to predicting reading difficulty. In *Proc. of HLT-NAACL 2004*, pages 193–200.

Fry, E. 1990. A readability formula for short passages. *Journal of Reading*, 33(8):594–597.

Gunning, R. 1952. *The Technique of Clear Writing*. McGraw-Hill, Cambridge, MA.

Heilman, Michael, Kevyn Collins-Thompson, Jamie Callan, and Maxine Eskenazi. 2007. Combining lexical and grammatical features to improve readability measures for first and second language texts. In *Proc. of NAACL-HLT 2007*, pages 460–467, Rochester, New York, April.

Heilman, Michael, Kevyn Collins-Thompson, and Maxine Eskenazi. 2008. An analysis of statistical models and features for reading difficulty prediction. In *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, pages 71–79, Columbus, Ohio, June. Association for Computational Linguistics.

Kanungo, Tapas and David Orr. 2009. Predicting the readability of short web summaries. In *Proc. of WSDM 2009*, pages 202–211, Barcelona, Spain, February.

Kincaid, J. P., R. P. Fishburne, R. L. Rogers, and B.S. Chissom. 1975. Derivation of new readability formulas for navy enlisted personnel. Technical Report Research Branch Report 8-75, Millington, TN: Naval Air Station.

Kneser, Reinhard and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proc. of ICASSP-95*, pages 181–184.

McCallum, Andrew and Kamal Nigam. 1998. A comparison of event models for naive Bayes text classification. In *Papers from the AAAI-98 Workshop on Text Categorization*, pages 41–48, Madison, WI, July.

McCord, Michael C. 1989. Slot grammar: A system for simpler construction of practical natural language grammars. In *Proceedings of the International Symposium on Natural Language and Logic*, pages 118–145, May.

McLaughlin, G. H. 1969. Smog: Grading: A new readability formula. *Journal of Reading*, 12:639–646.

Pitler, Emily and Ani Nenkova. 2008. Revisiting readability: A unified framework for predicting text quality. In *Proc. of EMNLP 2008*, pages 186–195, Waikiki,Honolulu,Hawaii, October.

Quinlan, J. R. 1987. Simplifying decision trees. *International Journal of Man-Machine Studies*, 27:221–234.

Rasmussen, Carl and Christopher Williams. 2006. *Gaussian Processes for Machine Leanring*. MIT Press, Cambridge, MA.

Riloff, E. and W. Phillips. 2004. An introduction to the Sundance and Autoslog systems. Technical Report UUCS-04-015, University of Utah School of Computing.

Riloff, Ellen. 1996. Automatically generating extraction patterns from untagged text. In *Proc. of 13th Natl. Conf. on Artificial Intelligence (AAAI-96)*, pages 1044–1049, Portland, OR.

Schwarm, Sarah E. and Mari Ostendorf. 2005. Reading level assessment using support vector machines and statistical language models. In *Proc. of ACL 2005*, pages 523–530, Ann Arbor, Michigan.

Si, Luo and James P. Callan. 2001. A statistical model for scientific readability. In *Proc. of CIKM 2001*, pages 574–576.

Smola, Alex J. and Bernhard Scholkopf. 1998. A tutorial on support vector regression. Technical Report NC2-TR-1998-030, NeuroCOLT2.

Stanley, Chen and Joshua Goodman. 1996. An empirical study of smoothing techniques for language modeling. In *Proc. of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-96)*, pages 310–318.

Stenner, A. J., I. Horabin, D. R. Smith, and M. Smith. 1988. *The Lexile Framework*. Durham, NC: MetaMetrics.

Thelen, M. and E. Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proc. of EMNLP 2002*, Philadelphia, PA, July.

Yang, Yiming and Xin Liu. 1999. A re-examination of text cateogrization methods. In *Proc. of 22nd Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 42–48, Berkeley, CA.

Value for Money: Balancing Annotation Effort, Lexicon Building and Accuracy for Multilingual WSD

Mitesh M. Khapra **Saurabh Sohoney** **Anup Kulkarni** **Pushpak Bhattacharyya**
Department of Computer Science and Engineering,
Indian Institute of Technology Bombay
{miteshk,saurabhsohoney,anup,pb}@cse.iitb.ac.in

Abstract

Sense annotation and lexicon building are costly affairs demanding prudent investment of resources. Recent work on multilingual WSD has shown that it is possible to leverage the annotation work done for WSD of one language (S_L) for another (T_L), by projecting Wordnet and sense marked corpus parameters of S_L to T_L. However, this work does not take into account the cost of manually cross-linking the words within aligned synsets. Further, it does not answer the question of *"Can better accuracy be achieved if a user is willing to pay additional money?"* We propose a measure for ***cost-benefit analysis*** which measures the *"value for money"* earned in terms of accuracy by investing in annotation effort and lexicon building. Two key ideas explored in this paper are (i) the use of ***probabilistic cross-linking model*** to reduce manual cross-linking effort and (ii) the use of ***selective sampling*** to inject a few training examples for hard-to-disambiguate words from the target language to boost the accuracy.

1 Introduction

Word Sense Disambiguation (WSD) is one of the most widely investigated problems of Natural Language Processing (NLP). Previous works have shown that supervised approaches to Word Sense Disambiguation which rely on sense annotated corpora (Ng and Lee, 1996; Lee et al., 2004) outperform unsupervised (Veronis, 2004) and knowledge based approaches (Mihalcea, 2005). However, creation of sense marked corpora has always remained a costly proposition, especially for some of the resource deprived languages.

To circumvent this problem, Khapra et al. (2009) proposed a WSD method that can be applied to a language even when no sense tagged corpus for that language is available. This is achieved by ***projecting Wordnet and corpus parameters*** from another language to the language in question. The approach is centered on a novel synset based multilingual dictionary (Mohanty et al., 2008) where the synsets of different languages are aligned and thereafter the words within the synsets are manually cross-linked. For example, the word W_{L_1} belonging to synset S of language L_1 will be manually cross-linked to the word W_{L_2} of the corresponding synset in language L_2 to indicate that W_{L_2} is the best substitute for W_{L_1} according to an experienced bilingual speaker's intuition.

We extend their work by addressing the following question on the economics of annotation, lexicon building and performance:

- *Is there an optimal point of balance between the annotation effort and the lexicon building (i.e. manual cross-linking) effort at which one can be assured of best value for money in terms of accuracy?*

To address the above question we first propose a probabilistic cross linking model to eliminate the effort of manually cross linking words within the source and target language synsets and calibrate the resultant trade-off in accuracy. Next, we show that by injecting examples for most frequent hard-to-disambiguate words from the target domain one can achieve higher accuracies at optimal

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 555–563,
Beijing, August 2010

cost of annotation. Finally, we propose a measure for ***cost-benefit analysis*** which identifies the optimal point of balance between these three related entities, viz., cross-linking, sense annotation and accuracy of disambiguation.

The remainder of this paper is organized as follows. In section 2 we present related work. In section 3 we describe the Synset based multilingual dictionary which enables parameter projection. In section 4 we discuss the work of Khapra et al. (2009) on parameter projection for multilingual WSD. Section 5 is on the economics of multilingual WSD. In section 6 we propose a probabilistic model for representing the cross-linkage of words within synsets. In section 7 we present a strategy for injecting hard-to-disambiguate cases from the target language using selective sampling. In section 8 we introduce a measure for **cost-benefit analysis** for calculating the value for money in terms of accuracy, annotation effort and lexicon building effort. In section 9 we describe the experimental setup. In section 10 we present the results followed by discussion in section 11. Section 12 concludes the paper.

2 Related Work

Knowledge based approaches to WSD such as Lesk's algorithm (Lesk, 1986), Walker's algorithm (Walker and Amsler, 1986), Conceptual Density (Agirre and Rigau, 1996) and PageRank (Mihalcea, 2005) are less demanding in terms of resources but fail to deliver good results. Supervised approaches like SVM (Lee et al., 2004) and k-NN (Ng and Lee, 1996), on the other hand, give better accuracies, but the requirement of large annotated corpora renders them unsuitable for resource scarce languages.

Recent work by Khapra et al. (2009) has shown that it is possible to project the parameters learnt from the annotation work of one language to another language provided aligned Wordnets for two languages are available. However, their work does not address the question of further improving the accuracy of WSD by using a small amount of training data from the target language. Some similar work has been done in the area of domain adaptation where Chan et al. (2007) showed that adding just 30% of the target data to the source data achieved the same performance as that obtained by taking the entire source and target data. Similarly, Agirre and de Lacalle (2009) reported a 22% error reduction when source and target data were combined for training a classifier, compared to the case when only the target data was used for training the classifier. However, such combining of training statistics has not been tried in cases where the source data is in one language and the target data is in another language.

To the best of our knowledge, no previous work has attempted to perform resource conscious **all-words multilingual Word Sense Disambiguation** by finding a trade-off between the cost (in terms of annotation effort and lexicon creation effort) and the quality in terms of F-score.

3 Synset based multilingual dictionary

A novel and effective method of storage and use of dictionary in a multilingual setting was proposed by Mohanty et al. (2008). For the purpose of current discussion, we will refer to this multilingual dictionary framework as *MultiDict*. One important departure in this framework from the traditional dictionary is that **synsets are linked, and after that the words inside the synsets are linked**. The basic mapping is thus between synsets and thereafter between the words.

Concepts	L1 (English)	L2 (Hindi)	L3 (Marathi)
04321: a youthful male person	{malechild, boy}	{लड़का *(ladkaa)*, बालक *(baalak)*, बच्चा *(bachchaa)*}	{मुलगा *(mulgaa)*, पोरगा *(porgaa)*, पोर *(por)*}

Table 1: Multilingual Dictionary Framework

Table 1 shows the structure of MultiDict, with one example row standing for the concept of *boy*. The first column is the pivot describing a concept with a unique ID. The subsequent columns show the words expressing the concept in respective languages (in the example table, *English, Hindi and Marathi*). After the synsets are linked, cross linkages are set up manually from the words of a synset to the words of a linked synset of the pivot language. For example, for the Marathi word मुलगा *(mulgaa)*, "a youthful male person", the correct lexical substitute from the corresponding Hindi synset is *लड़का (ladkaa)*. The average number of such links per synset per language pair is approximately 3.

4 Parameter Projection

Khapra et al. (2009) proposed that the various parameters essential for domain-specific Word Sense Disambiguation can be broadly classified into two categories:

Wordnet-dependent parameters:

- belongingness-to-dominant-concept
- conceptual distance
- semantic distance

Corpus-dependent parameters:

- sense distributions
- corpus co-occurrence

They proposed a scoring function (Equation (1)) which combines these parameters to identify the correct sense of a word in a context:

$$S^* = \arg\max_i(\theta_i V_i + \sum_{j \in J} W_{ij} * V_i * V_j) \qquad (1)$$

where,

$$\begin{aligned} i &\in \textit{Candidate Synsets} \\ J &= \textit{Set of disambiguated words} \\ \theta_i &= BelongingnessToDominantConcept(S_i) \\ V_i &= P(S_i|word) \\ W_{ij} &= CorpusCooccurrence(S_i, S_j) \\ &\quad * 1/WNConceptualDistance(S_i, S_j) \\ &\quad * 1/WNSemanticGraphDistance(S_i, S_j) \end{aligned}$$

The first component $\theta_i V_i$ of Equation (1) captures influence of the corpus specific sense of a word in a domain. The other component $W_{ij} * V_i * V_j$ captures the influence of interaction of the candidate sense with the senses of context words weighted by factors of co-occurrence, conceptual distance and semantic distance.

Wordnet-dependent parameters depend on the structure of the Wordnet whereas the ***Corpus-dependent parameters*** depend on various statistics learnt from a sense marked corpora. Both the tasks of (a) constructing a Wordnet from scratch and (b) collecting sense marked corpora for multiple languages are tedious and expensive. Khapra et al. (2009) observed that by ***projecting relations*** from the Wordnet of a language and by ***projecting corpus statistics*** from the sense marked corpora of the language to those of the target language, *the effort required in constructing semantic graphs for multiple Wordnets and collecting sense marked corpora for multiple languages can be avoided or reduced.* At the heart of their work lies the *MultiDict* described in previous section which facilitates parameter projection in the following manner:

1. By linking with the synsets of a pivot resource rich language (Hindi, in our case), the cost of building Wordnets of other languages is partly reduced (semantic relations are inherited). The Wordnet parameters of Hindi Wordnet now become projectable to other languages.

2. For calculating corpus specific sense distributions, $P(Sense\ S_i|Word\ W)$, we need the counts, $\#(S_i, W)$. By using cross linked words in the synsets, these counts become projectable to the target language (Marathi, in our case) as they can be approximated by the counts of the cross linked Hindi words calculated from the Hindi sense marked corpus as follows:

$$P(S_i|W) = \frac{\#(S_i, marathi\_word)}{\sum_j \#(S_j, marathi\_word)}$$

$$P(S_i|W) \approx \frac{\#(S_i, cross\_linked\_hindi\_word)}{\sum_j \#(S_j, cross\_linked\_hindi\_word)}$$

The rationale behind the above approximation is the observation that within a domain sense distributions remain the same across languages.

5 The Economics of Multilingual WSD

The problem of multilingual WSD using parameter projection can be viewed as an economic system consisting of three factors. The first factor is the cost of manually cross-linking the words in a synsets of the target language to the words in the corresponding synset in the pivot language. The second factor is the cost of sense annotated data from the target language. The third factor is the accuracy of WSD The first two factors in some sense relate to the cost of purchasing a commodity and the third factor relates to the commodity itself.

The work of Khapra et al. (2009) as described above does not attempt to reach an optimal cost-benefit point in this economic system. They place their bets on manual cross-linking only and settle for the accuracy achieved thereof. Specifically, they do not explore the inclusion of small amount of annotated data from the target language to boost the accuracy (as mentioned earlier, supervised systems which use annotated data from the target language are known to perform better). Further, it is conceivable that with respect to accuracy-cost trade-off, there obtains a case for *balancing* one cost against the other, *viz.*, the cost of cross-linking and the cost of annotation. In some cases bilingual lexicographers (needed for manual cross-linking) may be more expensive compared to monolingual annotators. There it makes sense to place fewer bets on manual cross-linking and more on collecting annotated corpora. On the other hand if manual cross-linking is cheap then a very small amount of annotated corpora can be used in conjunction with full manual cross-linking to boost the accuracy. Based on the above discussion, if k_a is the cost of sense annotating one word, k_c is the cost of manually cross-linking a word and A is the accuracy desired then the problem of multilingual WSD can be cast as an optimization problem:

$$\begin{aligned} &minimize \quad w_a * k_a + w_c * k_c \\ &s.t. \\ &\qquad Accuracy \geq A \end{aligned}$$

where, w_c and w_a are the number of words to be manually cross linked and annotated respectively. Ours is thus a 3-factor economic model (cross-linking, annotation and accuracy) as opposed to the 2-factor model (cross-linking, accuracy) proposed by Khapra et al. (2009).

6 Optimal cross-linking

As mentioned earlier, in some cases where bilingual lexicographers are expensive we might be interested in reducing the effort of manual cross-linking. For such situations, we propose that only a small number of words, comprising of the most frequently appearing ones should be manually cross linked and the rest of the words should be cross-linked using a probabilistic model. The rationale here is simple: invest money in words which are bound to occur frequently in the test data and achieve maximum impact on the accuracy. In the following paragraphs, we explain our probabilistic cross linking model.

The model proposed by Khapra et al. (2009) is a deterministic model where the expected count for (Sense S, Marathi_Word W), *i.e.*, the number of times the word W appears in sense S is approximated by the count for the corresponding cross linked Hindi word. Such a model assumes that each Marathi word links to appropriate Hindi word(s) as identified manually by a lexicographer. Instead, **we propose a probabilistic model where a Marathi word can link to every word in the corresponding Hindi synset with some probability**. The expected count for (S, W) can then be estimated as:

$$E[\#(S,W)] = \sum_{h_i \in cross\_links} P(h_i|W,S) * \#(S,h_i) \quad (2)$$

where, $P(h_i|W,S)$ is the probability that the word h_i from the corresponding Hindi synset is the correct cross-linked word for the given Marathi word. For example, one of the senses of the Marathi word *maan* is {neck} *i.e.* *"the body part which connects the head to the rest of the body"*. The corresponding Hindi synset has 10 words {*gardan, gala, greeva, halak, kandhar and so on*}. Thus, using Equation (2), the expected count, $E[C(\{neck\}, maan)]$, is calculated as:

$$\begin{aligned} &E[\#(\{neck\}, maan)] = \\ &\quad P(gardan|maan, \{neck\}) * \#(\{neck\}, gardan) \\ &+ P(gala|maan, \{neck\}) * \#(\{neck\}, gala) \\ &+ P(greeva|maan, \{neck\}) * \#(\{neck\}, greeva) \\ &+ \ \ldots \ so\ on\ for\ all\ words\ in\ the\ Hindi\ synset \end{aligned}$$

Instead of using a uniform probability distribution over the Hindi words we go by the empirical observation that some words in a synset are more representative of that sense than other words, *i.e. some words are more preferred while expressing that sense*. For example, out of the 10 words in the Hindi synset only 2 words {*gardan, gala*} appeared in the corpus. We thus estimate the value of $P(h_i|W, S)$ empirically from the Hindi sense marked corpus by making the following independence assumption:

$$P(h_i|W, S) = P(h_i|S)$$

The rationale behind the above independence assumption becomes clear if we represent words and synsets using the Bayesian network of Figure 1. Here, the Hindi word h_i and the Marathi word W

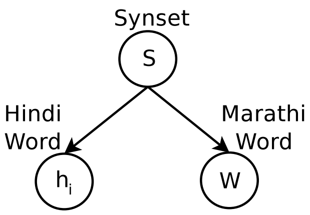


Figure 1: Bayesian network formed by a synset S and the constituent Hindi and Marathi words

are considered to be derived from the same parent concept S. In other words, they represent two different manifestations- one in Hindi and one in Marathi- of the same synset S. Given the above representation, it is easy to see that given the parent synset S, the Hindi word h_i is independent of the Marathi word W.

7 Optimal annotation using Selective Sampling

In the previous section we dealt with the question of optimal cross-linking. Now we take up the other dimension of this economic system, *viz.*, optimal use of annotated corpora for better accuracy. In other words, if an application demands higher accuracy for WSD and is willing to pay for some annotation then there should be a way of ensuring best possible accuracy at lowest possible cost. This can be done by including small amount of sense annotated data from the target language. The simplest strategy is to randomly annotate text from the target language and use it as training data. However, this strategy of random sampling may not be the most optimum in terms of cost. Instead, we propose a selective sampling strategy where the aim is to identify ***hard-to-disambiguate*** words from the target language and use them for training.

The algorithm proceeds as follows:

1. First, using the probabilistic cross linking model and aligned Wordnets we learn the parameters described in Section 4.

2. We then apply this scoring function on untagged examples (development set) from the target language and identify ***hard-to-disambiguate*** words *i.e.*, the words which were disambiguated with a very low confidence.

3. Training instances of these words are then injected into the training data and the parameters learnt from them are used instead of the projected parameters learnt from the source language corpus.

Thus, the selective sampling strategy ensures that we get maximum value for money by spending it on annotating only those words which would otherwise not have been disambiguated correctly. A random selection strategy, in contrast, might bring in words which were disambiguated correctly using only the projected parameters.

8 A measure for cost-benefit analysis

We need a measure for cost-benefit analysis based on the three dimensions of our economic system, *viz.*, annotation effort, lexicon creation effort and performance in terms of F-score. The first two dimensions can be fused into a single dimension by expressing the annotation effort and lexicon creation effort in terms of cost incurred. For example, we assume that the cost of annotating one word is k_a and the cost of cross-linking one word is k_c rupees. Further, we define a baseline and an upper bound for the F-score. In this case, the baseline would be the accuracy that can be obtained without spending any money on cross-linking and annotation in the target language. An upper bound could be the best F-score obtained using a large amount of annotated corpus in the target domain. Based on the above description, an ideal measure for cost-benefit analysis would assign a

1. reward depending on the improvement over the baseline performance.

2. penalty depending on the difference from the upper bound on performance.

3. reward inversely proportional to the cost incurred in terms of annotation effort and/or manual cross-linking.

Based on the above wish-list we propose a measure for cost-benefit analysis. Let,

$$
\begin{aligned}
MGB &= Marginal\ Gain\ over\ Baseline\ (MGB) \\
&= \frac{Performance(P) - Baseline(B)}{Cost(C)} \\
MDU &= Marginal\ Drop\ from\ Upperbound\ (MDU) \\
&= \frac{UpperBound(U) - Performance(P)}{Cost(C)}
\end{aligned}
$$

then

$$CostBenefit(CB) = MGB - MDU$$

9 Experimental Setup

We used Hindi as the source language (S_L) and trained a WSD engine using Hindi sense tagged corpus. The parameters thus learnt were then projected using the *MultiDict* (refer section 3 and 4) to build a resource conscious Marathi (T_L) WSD engine. We used the same dataset as described in Khapra et al. (2009) for all our experiments. The data was collected from two domains, *viz.*, Tourism and Health. The data for Tourism domain was collected by manually translating English documents downloaded from Indian Tourism websites into Hindi and Marathi. Similarly, English documents for Health domain were obtained from two doctors and were manually translated into Hindi and Marathi. The Hindi and Marathi documents thus created were manually sense annotated by two lexicographers adept in Hindi and Marathi using the respective Wordnets as sense repositories. Table 2 summarizes some statistics about the corpora.

As for cross-linking, Hindi is used as the pivot language and words in Marathi synset are linked to the words in the corresponding Hindi synset. The total number of cross-links that were manually setup were 3600 for Tourism and 1800 for Health. The cost of cross-linking as well as sense annotating one word was taken to be 10 rupees. These costs were estimated based on quotations from lexicographers. However, these costs need to be taken as representative values only and may vary greatly depending on the availability of skilled bilingual lexicographers and skilled monolingual annotators.

Language	#of polysemous words		average degree of polysemy	
	Tourism	Health	Tourism	Health
Hindi	56845	30594	3.69	3.59
Marathi	34156	10337	3.41	3.60

Table 2: Number of polysemous words and average degree of polysemy.

10 Results

Tables 3 and 4 report the average 4-fold performance on Marathi Tourism and Health data using different proportions of available resources, *i.e.*, annotated corpora and manual cross-links. In each of these tables, along the rows, we increase the amount of Marathi sense annotated corpora from 0K to 6K. Similarly, along the columns we show the increase in the number of manual cross links (MCL) used. For example, the second column of Tables 3 and 4 reports the F-scores when probabilistic cross-linking (PCL) was used for all words (*i.e.*, no manual cross-links) and varying amounts of sense annotated corpora from Marathi were used. Similarly, the first row represents the case in which no sense annotated corpus from Marathi was used and varying amounts of manual cross-links were used.

We report three values in the tables, *viz.*, F-score (F), cost in terms of money (C) and the cost-benefit (CB) obtained by using x amount of annotated corpus and y amount of manual cross-links. The cost was estimated using the values given in section 9 (*i.e.*, 10 rupees for cross-linking or sense annotating one word). For calculating, the cost-benefit baseline was taken as the F-score obtained by using no cross-links and no annotated corpora *i.e.* 68.21% for Tourism and 67.28% for Health (see first F-score cell of Tables 3 and 4). Similarly the upper bound (F-scores obtained by training on entire Marathi sense marked corpus) for Tourism and Health were 83.16% and 80.67% respectively (see last row of Table 5).

Due to unavailability of large amount of tagged Health corpus, the injection size was varied from 0-to-4K only. In the other dimension, we varied the cross-links from 0 to 1/3rd to 2/3rd to full only

Selective Sampling	Only PCL			1/3 MCL			2/3 MCL			Full MCL		
	F	C	CB	F	C	CB	F	C	CB	F	C	CB
0K	68.21	0	-	72.08	12	-0.601	73.31	24	-0.198	73.34	36	-0.130
1K	71.18	10	-0.901	74.96	22	-0.066	77.58	34	0.111	77.73	46	0.089
2K	74.35	20	-0.134	76.96	32	0.080	**78.57**	**44**	**0.131**	79.23	56	0.127
3K	75.21	30	-0.032	77.78	42	0.100	78.68	54	0.111	79.8	66	0.125
4K	76.40	40	0.036	78.66	52	0.114	79.18	64	0.110	80.36	76	0.123
5K	77.04	50	0.054	78.51	62	0.091	79.60	74	0.106	80.46	86	0.111
6K	78.58	60	0.097	79.75	72	0.113	80.8	84	0.122	80.44	96	0.099

Table 3: F-Score (F) in %, Cost (C) in thousand rupees and Cost Benefit (CB) values using different amounts of sense annotated corpora and manual cross links in Tourism domain.

Selective Sampling	Only PCL			1/3 MCL			2/3 MCL			Full MCL		
	F	C	CB	F	C	CB	F	C	CB	F	C	CB
0K	67.28	0	-	71.39	6	-0.862	73.06	12	-0.153	73.34	18	-0.071
1K	72.51	10	-0.293	75.57	16	0.199	**77.41**	**22**	**0.312**	78.16	28	0.299
2K	75.64	20	0.167	77.29	26	0.255	78.13	32	0.260	78.63	38	0.245
3K	76.78	30	0.187	79.35	36	0.299	79.79	42	0.277	79.88	48	0.246
4K	77.42	40	0.172	79.59	46	0.244	80.54	52	0.253	80.15	58	0.213

Table 4: F-Score (F) in %, Cost (C) in thousand rupees and Cost Benefit (CB) values using different amounts of sense annotated corpora and manual cross links in Health domain.

Strategy	Tourism	Health
WFS	57.86	52.77
Only PCL	68.21	67.28
1/6 MCL	69.95	69.57
2/6 MCL	72.08	71.39
3/6 MCL	72.97	72.61
4/6 MCL	73.39	73.06
5/6 MCL	73.55	73.27
Full MCL	73.62	73.34
Upper Bound	83.16	80.67

Table 5: F-score (in %) obtained by using different amounts of manually cross linked words

Strategy	Size of target side annotated corpus						
	0K	1K	2K	3K	4K	5K	6K
Random + PCL	68.21	70.62	71.79	73.03	73.61	76.42	77.52
Random + MCL	73.34	75.32	75.89	76.79	76.83	78.91	80.87
Selective Sampling + PCL	68.21	71.18	74.35	75.21	76.40	77.04	78.58
Selective Sampling + MCL	73.34	77.73	79.23	79.8	79.8	80.46	80.44

Table 6: Comparing F-scores obtained using random sampling and selective sampling (Tourism)

Strategy	Size of target side annotated corpus						
	0K	1K	2K	3K	4K	5K	6K
Annotation + PCL	68.21	71.20	74.35	75.21	76.40	77.04	78.58
Only Annotation	57.86	62.32	64.84	66.86	68.89	69.64	71.82

Table 7: Comparing F-scores obtained using Only Annotation and Annotation + PCL(Tourism)

(refer to Tables 3 and 4). However, to give an idea about the soundness of probabilistic cross-linking we performed a separate set of experiments by varying the number of cross-links and using no sense annotated corpora. Table 5 summarizes these results and compares them with the baseline (Wordnet first sense) and skyline.

In Table 6 we compare our selective sampling strategy with random sampling when fully probabilistic cross-linking (PCL) is used and when fully manual cross-linking (MCL) is used. Here again, due to lack of space we report results only on Tourism domain. However, we would like to mention that similar experiments on Health domain showed that the results were indeed consistent.

Finally, in Table 7 we compare the accuracies obtained when certain amount of annotated corpus from Marathi is used alone, with the case when the same amount of annotated corpus is used in conjunction with probabilistic cross-linking. While calculating the results for the second row in Table 7, we found that the recall was very low due to the small size of injections. Hence, to ensure a fair comparison with our strategy (first row) we used the Wordnet first sense (WFS) for these recall errors (a typical practice in WSD literature).

11 Discussions

We make the following observations:

1. PCL v/s MCL: Table 5 shows that the probabilistic cross-linking model performs much better than the WFS (a typically reported baseline) and it comes very close to the performance of manual cross-linking. This establishes the soundness of the probabilistic model and suggests that with a little compromise in the accuracy, the model can be used as an approximation to save the cost of manual cross-linking. Further, in Table 7 we see that when PCL is used in conjunction with certain amount of annotated corpus we get up to 9% improvement in F-score as compared to the case when the same amount of annotated corpus is used alone. Thus, in the absence of skilled bilingual lexicographers, PCL can still be used to boost the accuracy obtained using annotated corpora.

2. Selective Sampling v/s Random Annotation: Table 6 shows the benefit of selective sampling over random annotation. This benefit is felt more when the amount of training data injected from Marathi is small. For example, when an annotated corpus of size 2K is used, selective sampling gives an advantage of 3% to 4% over random selection. Thus the marginal gain (*i.e.*, value for money) obtained by using selective sampling is more than that obtained by using random annotation.

3. Optimal cost-benefit: Finally, we address the main message of our work, *i.e.*, finding the best cost benefit. By referring to Tables 3 and 4, we see that the best value for money in Tourism domain is obtained by manually cross-linking 2/3rd of all corpus words and sense annotating 2K target words and in the Health domain it is obtained by manually cross-linking 2/3rd of all corpus words but sense annotating only 1K words. This suggests that striking a balance between cross-linking and annotation gives the best value for money. Further, we would like to highlight that our 3-factor economic model is able to capture these relations better than the 2-factor model of Khapra et al. (2010). As per their model the best F-score achieved using manual cross-linking for ALL words was 73.34% for both Tourism and Health domain at a cost of 36K and 18K respectively. On the other hand, using our model we obtain higher accuracies of 76.96% in the Tourism domain (using 1/3rd manual cross-links and 2K injection) at a lower total cost (32K rupees) and 75.57% in the Health domain (using only 1/3rd cross-linking and 1K injection) at a lower cost (16K rupees).

12 Conclusion

We reported experiments on multilingual WSD using different amounts of annotated corpora and manual cross-links. We showed that there exists some trade-off between the accuracy and *balancing* the cost of annotation and lexicon creation. In the absence of skilled bilingual lexicographers one can use a probabilistic cross-linking model and still obtain good accuracies. Also, while sense annotating a corpus, careful selection of words using selective sampling can give better marginal gain as compared to random sampling.

References

Agirre, Eneko and Oier Lopez de Lacalle. 2009. Supervised domain adaption for wsd. In *EACL '09: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 42–50, Morristown, NJ, USA. Association for Computational Linguistics.

Agirre, Eneko and German Rigau. 1996. Word sense disambiguation using conceptual density. In *In Proceedings of the 16th International Conference on Computational Linguistics (COLING)*.

Chan, Y.S., H. T. Ng, and D. Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *In Proc. of ACL*.

Khapra, Mitesh M., Sapan Shah, Piyush Kedia, and Pushpak Bhattacharyya. 2009. Projecting parameters for multilingual word sense disambiguation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 459–467, Singapore, August. Association for Computational Linguistics.

Khapra, Mitesh, Sapan Shah, Piyush Kedia, and Pushpak Bhattacharyya. 2010. Domain-specific word sense disambiguation combining corpus based and wordnet based parameters. In *5th International Conference on Global Wordnet (GWC2010)*.

Lee, Yoong Keok, Hwee Tou Ng, and Tee Kiah Chia. 2004. Supervised word sense disambiguation with support vector machines and multiple knowledge sources. In *Proceedings of Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 137–140.

Lesk, Michael. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *In Proceedings of the 5th annual international conference on Systems documentation*.

Mihalcea, Rada. 2005. Large vocabulary unsupervised word sense disambiguation with graph-based algorithms for sequence data labeling. In *In Proceedings of the Joint Human Language Technology and Empirical Methods in Natural Language Processing Conference (HLT/EMNLP)*, pages 411–418.

Mohanty, Rajat, Pushpak Bhattacharyya, Prabhakar Pande, Shraddha Kalele, Mitesh Khapra, and Aditya Sharma. 2008. Synset based multilingual dictionary: Insights, applications and challenges. In *Global Wordnet Conference*.

Ng, Hwee Tou and Hian Beng Lee. 1996. Integrating multiple knowledge sources to disambiguate word senses: An exemplar-based approach. In *In Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 40–47.

Veronis, Jean. 2004. Hyperlex: Lexical cartography for information retrieval. In *Computer Speech and Language*, pages 18(3):223–252.

Walker, D. and R. Amsler. 1986. The use of machine readable dictionaries in sublanguage analysis. In *In Analyzing Language in Restricted Domains, Grishman and Kittredge (eds), LEA Press*, pages 69–83.

A Cross-lingual Annotation Projection Approach for Relation Detection

Seokhwan Kim[†], Minwoo Jeong[‡], Jonghoon Lee[†], Gary Geunbae Lee[†]
[†]Department of Computer Science and Engineering,
Pohang University of Science and Technology
{megaup|jh21983|gblee}@postech.ac.kr
[‡]Saarland University
m.jeong@mmci.uni-saarland.de

Abstract

While extensive studies on relation extraction have been conducted in the last decade, statistical systems based on supervised learning are still limited because they require large amounts of training data to achieve high performance. In this paper, we develop a cross-lingual annotation projection method that leverages parallel corpora to bootstrap a relation detector without significant annotation efforts for a resource-poor language. In order to make our method more reliable, we introduce three simple projection noise reduction methods. The merit of our method is demonstrated through a novel Korean relation detection task.

1 Introduction

Relation extraction aims to identify semantic relations of entities in a document. Many relation extraction studies have followed the Relation Detection and Characterization (RDC) task organized by the Automatic Content Extraction project (Doddington et al., 2004) to make multilingual corpora of English, Chinese and Arabic. Although these datasets encourage the development and evaluation of statistical relation extractors for such languages, there would be a scarcity of labeled training samples when learning a new system for another language such as Korean. Since manual annotation of entities and their relations for such *resource-poor languages* is very expensive, we would like to consider instead a weakly-supervised learning technique in order to learn the relation extractor without significant annotation efforts. To do this, we propose to leverage parallel corpora to project the relation annotation on the source language (e.g. English) to the target (e.g. Korean).

While many supervised machine learning approaches have been successfully applied to the RDC task (Kambhatla, 2004; Zhou et al., 2005; Zelenko et al., 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005; Zhang et al., 2006), few have focused on weakly-supervised relation extraction. For example, (Zhang, 2004) and (Chen et al., 2006) utilized weakly-supervised learning techniques for relation extraction, but they did not consider weak supervision in the context of cross-lingual relation extraction. Our key hypothesis on the use of parallel corpora for learning the relation extraction system is referred to as *cross-lingual annotation projection*. Early studies of cross-lingual annotation projection were accomplished for lexically-based tasks; for example part-of-speech tagging (Yarowsky and Ngai, 2001), named-entity tagging (Yarowsky et al., 2001), and verb classification (Merlo et al., 2002). Recently, there has been increasing interest in applications of annotation projection such as dependency parsing (Hwa et al., 2005), mention detection (Zitouni and Florian, 2008), and semantic role labeling (Pado and Lapata, 2009). However, to the best of our knowledge, no work has reported on the RDC task.

In this paper, we apply a cross-lingual annotation projection approach to binary *relation detection*, a task of identifying the relation between two entities. A simple projection method propagates the relations in source language sentences to

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 564–571,
Beijing, August 2010

word-aligned target sentences, and a target relation detector can bootstrap from projected annotation. However, this automatic annotation is unreliable because of mis-classification of source text and word alignment errors, so it causes a critical falling-off in annotation projection quality. To alleviate this problem, we present three noise reduction strategies: a heuristic filtering; an alignment correction with dictionary; and an instance selection based on assessment, and combine these to yield a better result.

We provide a quantitive evaluation of our method on a new Korean RDC dataset. In our experiment, we leverage an English-Korean parallel corpus collected from the Web, and demonstrate that the annotation projection approach and noise reduction method are beneficial to build an initial Korean relation detection system. For example, the combined model of three noise reduction methods achieves F1-scores of 36.9% (59.8% precision and 26.7% recall), favorably comparing with the 30.5% shown by the supervised baseline.[1]

The remainder of this paper is structured as follows. In Section 2, we describe our cross-lingual annotation projection approach to relation detection task. Then, we present the noise reduction methods in Section 3. Our experiment on the proposed Korean RDC evaluation set is shown in Section 4 and Section 5, and we conclude this paper in Section 6.

2 Cross-lingual Annotation Projection for Relation Detection

The annotation projection from a resource-rich language L_1 to a resource-poor language L_2 is performed by a series of three subtasks: annotation, projection and assessment.

The annotation projection for relation detection can be performed as follows:

1) For a given pair of bi-sentences in parallel corpora between a resource-rich language L_1 and a target language L_2, the relation detection task is carried out for the sentence in L_1.

2) The annotations obtained by analyzing the sentence in L_1 are projected onto the sentence in L_2 based on the word alignment information.

3) The projected annotations on the sentence in L_2 are utilized as resources to perform the relation detection task for the language L_2.

2.1 Annotation

The first step to projecting annotations from L_1 onto L_2 is obtaining annotations for the sentences in L_1. Since each instance for relation detection is composed of a pair of entity mentions, the information about entity mentions on the given sentences should be identified first. We detect the entities in the L_1 sentences of the parallel corpora. Entity identification generates a number of instances for relation detection by coupling two entities within each sentence. For each instance, the existence of semantic relation between entity mentions is explored, which is called relation detection. We assume that there exist available models or systems for all annotation processes, including not only an entity tagger and a relation detector themselves, but also required preprocessors such as a part-of-speech tagger, base-phrase chunker, and syntax parser for analyzing text in L_1.

Figure 1 shows an example of annotation projection for relation detection of a bitext in English and Korean. The annotation of the sentence in English shows that "Jan Mullins" and "Computer Recycler Incorporated" are entity mentions of a person and an organization, respectively. Furthermore, the result indicates that the pair of entities has a semantic relationship categorized as "ROLE.Owner" type.

2.2 Projection

In order to project the annotations from the sentences in L_1 onto the sentences in L_2, we utilize the information of word alignment which plays an important role in statistical machine translation techniques. The word alignment task aims to identify translational relationships among the words in a bitext and produces a bipartite graph with a set of edges between words with translational relationships as shown in Figure 1. In the same manner as the annotation in L_1, entities are

[1] The dataset and the parallel corpus are available on the author's website, http://isoft.postech.ac.kr/∼megaup/research/resources/.

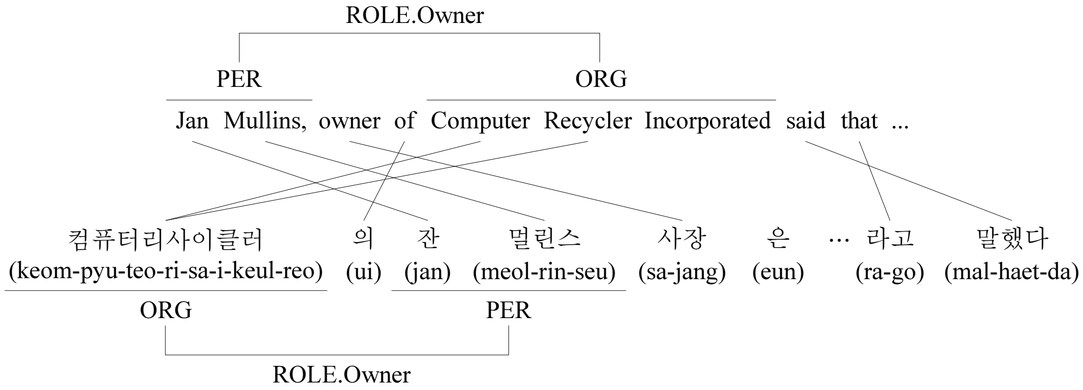


Figure 1: An example of annotation projection for relation detection of a bitext in English and Korean

considered as the first units to be projected. We assume that the words of the sentences in L_2 aligned with a given entity mention in L_1 inherit the information about the original entity in L_1.

After projecting the annotations of entity mentions, the projections for relational instances follow. A projection is performed on a projected instance in L_2 which is a pair of projected entities by duplicating annotations of the original instance in L_1.

Figure 1 presents an example of projection of a positive relational instance between "Jan Mullins" and "Computer Recycler Incorporated" in the English sentence onto its translational counterpart sentence in Korean. "Jan meol-rin-seu" and "keom-pyu-teo-ri-sa-i-keul-reo" are labeled as entity mentions with types of a person's name and an organization's name respectively. In addition, the instance composed of the two projected entities is annotated as a positive instance, because its original instance on the English sentence also has a semantic relationship.

As the description suggests, the annotation projection approach is highly dependant on the quality of word alignment. However, the results of automatic word alignment may include several noisy or incomplete alignments because of technical difficulties. We present details to tackle the problem by relieving the influence of alignment errors in Section 3.

2.3 Assessment

The most important challenge for annotation projection approaches is how to improve the robustness against the erroneous projections. The noise produced by not only word alignment but also mono-lingual annotations in L_1 accumulates and brings about a drastic decline in the quality of projected annotations.

The simplest policy of utilizing the projected annotations for relation detection in L_2 is to consider that all projected instances are equivalently reliable and to employ entire projections as training instances for the task without any filtering. In contrast with this policy, which is likely to be substandard, we propose an alternative policy where the projected instances are assessed and only the instances judged as reliable by the assessment are utilized for the task. Details about the assessment are provided in Section 3.

3 Noise Reduction Strategies

The efforts to reduce noisy projections are considered indispensable parts of the projection-based relation detection method in a resource-poor language. Our noise reduction approach includes the following three strategies: heuristic-based alignment filtering, dictionary-based alignment correction, and assessment-based instance selection.

3.1 Heuristic-based Alignment Filtering

In order to improve the performance of annotation projection approaches, we should break the bottleneck caused by the low quality of automatic word alignment results. As relation detection is carried out for each instance consisting of two entity mentions, the annotation projection for relation detection concerns projecting only entity mentions and their relational instances. Since this is different from other shallower tasks such as part-of-speech tagging, base phrase chunking, and dependency parsing which should consider projections for all word units, we define and apply some heuristics specialized to projections of entity mentions and relation instances to improve robustness of the method against erroneous alignments, as follows:

- A projection for an entity mention should be based on alignments between contiguous word sequences. If there are one or more gaps in the word sequence in L2 aligned with an entity mention in the sentence in L1, we assume that the corresponding alignments are likely to be erroneous. Thus, the alignments of non-contiguous words are excluded in projection.

- Both an entity mention in L_1 and its projection in L_2 should include at least one base noun phrase. If no base noun phrase occurs in the original entity mention in L_1, it may suggest some errors in annotation for the sentence in L_1. The same case for the projected instance raises doubts about alignment errors. The alignments between word sequences without any base noun phrase are filtered out.

- The projected instance in L2 should satisfy the clausal agreement with the original instance in L1. If entities of an instance are located in the same clause (or different clauses), its projected instance should be in the same manner. The instances without clausal agreement are ruled out.

3.2 Dictionary-based Alignment Correction

The errors in word alignment are composed of not only imprecise alignments but also incomplete alignments. If an alignment of an entity among two entities of a relation instance is not provided in the result of the word alignment task, the projection for the corresponding instance is unavailable. Unfortunately, the above-stated alignment filtering heuristics for improving the quality of projections make the annotation loss problems worse by filtering out several alignments likely to be noisy.

In order to solve this problem, a dictionary-based alignment correction strategy is incorporated in our method. The strategy requires a bilingual dictionary for entity mentions. Each entry of the dictionary is a pair of entity mention in L_1 and its translation or transliteration in L_2. For each entity to be projected from the sentence in L_1, its counterpart in L_2 is retrieved from the bilingual dictionary. Then, we seek the retrieved entity mention from the sentence in L_2 by finding the longest common subsequence. If a subsequence matched to the retrieved mention is found in the sentence in L_2, we make a new alignment between it and its original entity on the L_1 sentence.

3.3 Assessment-based Instance Selection

The reliabilities of instances projected via a series of independent modules are different from each other. Thus, we propose an assessment strategy for each projected instance. To evaluate the reliability of a projected instance in L_2, we use the confidence score of monolingual relation detection for the original counterpart instance in L_1. The acceptance of a projected instance is determined by whether the score of the instance is larger than a given threshold value θ. Only accepted instances are considered as the results of annotation projection and applied to solve the relation detection task in target language L_2.

4 Experimental Setup

To demonstrate the effectiveness of our cross-lingual annotation projection approach for relation detection, we performed an experiment on relation detection in Korean text with propagated annotations from English resources.

4.1 Annotation

The first step to evaluate our method was annotating the English sentences in a given parallel corpus. We use an English-Korean parallel corpus crawled from an English-Korean dictionary on the web. The parallel corpus consists of 454,315 bi-sentence pairs in English and Korean [2]. The English sentences in the parallel corpus were prepro-

[2]The parallel corpus collected and other resources are all available in our website
http://isoft.postech.ac.kr/∼megaup/research/resources/

cessed by the Stanford Parser [3] (Klein and Manning, 2003) which provides a set of analyzed results including part-of-speech tag sequences, a dependency tree, and a constituent parse tree for a sentence.

The annotation for English sentences is divided into two subtasks: entity mention recognition and relation detection. We utilized an off-the-shelf system, Stanford Named Entity Recognizer [4] (Finkel et al., 2005) for detecting entity mentions on the English sentences. The total number of English entities detected was 285,566. Each pair of recognized entities within a sentence was considered as an instance for relation detection.

A classification model learned with the training set of the ACE 2003 corpus which consists of 674 documents and 9,683 relation instances was built for relation detection in English. In our implementation, we built a tree kernel-based SVM model using SVM-Light [5] (Joachims, 1998) and Tree Kernel Tools [6] (Moschitti, 2006). The subtree kernel method (Moschitti, 2006) for shortest path enclosed subtrees (Zhang et al., 2006) was adopted in our model. Our relation detection model achieved 81.2/69.8/75.1 in Precision/Recall/F-measure on the test set of the ACE 2003 corpus, which consists of 97 documents and 1,386 relation instances.

The annotation of relations was performed by determining the existence of semantic relations for all 115,452 instances with the trained model for relation detection. The annotation detected 22,162 instances as positive which have semantic relations.

4.2 Projection

The labels about entities and relations in the English sentences of the parallel corpora were propagated into the corresponding sentences in Korean. The Korean sentences were preprocessed by our part-of-speech tagger [7] (Lee et al., 2002) and a dependency parser implemented by MSTParser with

[3]http://nlp.stanford.edu/software/lex-parser.shtml
[4]http://nlp.stanford.edu/software/CRF-NER.shtml
[5]http://svmlight.joachims.org/
[6]http://disi.unitn.it/∼moschitt/Tree-Kernel.htm
[7]http://isoft.postech.ac.kr/∼megaup/research/postag/

Filter	Without assessing	With assessing
none	97,239	39,203
+ heuristics	31,652	12,775
+ dictionary	39,891	17,381

Table 1: Numbers of projected instances

a model trained on the Sejong corpus (Kim, 2006).

The annotation projections were performed on the bi-sentences of the parallel corpus followed by descriptions mentioned in Section 2.2. The bi-sentences were processed by the GIZA++ software (Och and Ney, 2003) in the standard configuration in both English-Korean and Korean-English directions. The bi-direcional alignments were joined by the grow-diag-final algorithm, which is widely used in bilingual phrase extraction (Koehn et al., 2003) for statistical machine translation. This system achieved 65.1/41.6/50.8 in Precision/Recall/F-measure in our evaluation of 201 randomly sampled English-Korean bi-sentences with manually annotated alignments.

The number of projected instances varied with the applied strategies for reducing noise as shown in Table 1. Many projected instances were filtered out by heuristics, and only 32.6% of the instances were left. However, several instances were rescued by dictionary-based alignment correction and the number of projected instances increased from 31,652 to 39,891. For all cases of noise reduction strategies, we performed the assessment-based instance selection with a threshold value θ of 0.7, which was determined empirically through the grid search method. About 40% of the projected instances were accepted by instance selection.

4.3 Evaluation

In order to evaluate our proposed method, we prepared a dataset for the Korean RDC task. The dataset was built by annotating the information about entities and relations in 100 news documents in Korean. The annotations were performed by two annotators following the guidelines for the ACE corpus processed by LDC. Our Korean RDC corpus consists of 835 sentences, 3,331 entity mentions, and 8,354 relation instances. The sen-

Model	w/o assessing			with assessing		
	P	R	F	P	R	F
Baseline	60.5	20.4	30.5	-	-	-
Non-filtered	22.5	6.5	10.0	29.1	13.2	18.2
Heuristic	51.4	15.5	23.8	56.1	22.9	32.5
Heuristic + Dictionary	55.3	19.4	28.7	59.8	26.7	36.9

Table 2: Experimental Results

tences of the corpus were preprocessed by equivalent systems used for analyzing Korean sentences for projection. We randomly divided the dataset into two subsets with the same number of instances for use as a training set to build the baseline system and for evaluation.

For evaluating our approach, training instance sets to learn models were prepared for relation detection in Korean. The instances of the training set (half of the manually built Korean RDC corpufs) were used to train the baseline model. All other sets of instances include these baseline instances and additional instances propagated by the annotation projection approach. The training sets with projected instances are categorized into three groups by the level of applied strategies for noise reduction. While the first set included all projections without any noise reduction strategies, the second included only the instances accepted by the heuristics. The last set consisted of the results of a series of heuristic-based filtering and dictionary-based correction. For each training set with projected instances, an additional set was derived by performing assessment-based instance selection.

We built the relation detection models for all seven training sets (a baseline set, three projected sets without assessing, and three projected sets with assessing). Our implementations are based on the SVM-Light and Tree Kernel Tools described in the former subsection. The shortest path dependency kernel (Bunescu and Mooney, 2005) implemented by the subtree kernel method (Moschitti, 2006) was adopted to learn all models.

The performance for each model was evaluated with the predictions of the model on the test set, which was the other half of Korean RDC corpus. We measured the performances of the models on true entity mentions with true chaining of coreference. Precision, Recall and F-measure were adopted for our evaluation.

5 Experimental Results

Table 2 compares the performances of the different models which are distinguished by the applied strategies for noise reduction. It shows that:

- The model with non-filtered projections achieves extremely poor performance due to a large number of erroneous instances. This indicates that the efforts for reducing noise are urgently needed.
- The heuristic-based alignment filtering helps to improve the performance. However, it is much worse than the baseline performance because of a falling-off in recall.
- The dictionary-based correction to our projections increased both precision and recall compared with the former models with projected instances. Nevertheless, it still fails to achieve performance improvement over the baseline model.
- For all models with projection, the assessment-based instance selection boosts the performances significantly. This means that this selection strategy is crucial in improving the performance of the models by excluding unreliable instances with low confidence.
- The model with heuristics and assessments finally achieves better performance than the baseline model. This suggests that the projected instances have a beneficial influence

on the relation detection task when at least these two strategies are adopted for reducing noises.

- The final model incorporating all proposed noise reduction strategies outperforms the baseline model by 6 in F-measure. This is due to largely increased recall by absorbing more useful features from the well-refined set of projected instances.

The experimental results show that our proposed techniques effectively improve the performance of relation detection in the resource-poor Korean language with a set of annotations projected from the resource-rich English language.

6 Conclusion

This paper presented a novel cross-lingual annotation projection method for relation extraction in a resource-poor language. We proposed methods of propagating annotations from a resource-rich language to a target language via parallel corpora. In order to relieve the bad influence of noisy projections, we focused on the strategies for reducing the noise generated during the projection. We applied our methods to the relation detection task in Korean. Experimental results show that the projected instances from an English-Korean parallel corpus help to improve the performance of the task when our noise reduction strategies are adopted.

We would like to introduce our method to the other subtask of relation extraction, which is relation categorization. While relation detection is a binary classification problem, relation categorization can be solved by a classifier for multiple classes. Since the fundamental approaches of the two tasks are similar, we expect that our projection-based relation detection methods can be easily adapted to the relation categorization task.

For this further work, we are concerned about the problem of low performance for Korean, which was below 40 for relation detection. The relation categorization performance is mostly lower than detection because of the larger number of classes to be classified, so the performance of projection-based approaches has to be improved in order to apply them. An experimental result of this work shows that the most important factor in improving the performance is how to select the reliable instances from a large number of projections. We plan to develop more elaborate strategies for instance selection to improve the projection performance for relation extraction.

Acknowledgement

This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency) (NIPA-2010-C1090-1031-0009).

References

Bunescu, Razvan C. and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, page 724731.

Chen, Jinxiu, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. 2006. Relation extraction using label propagation based semi-supervised learning. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 129–136, Sydney, Australia. Association for Computational Linguistics.

Culotta, Aron and Jaffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of ACL*, volume 4.

Doddington, George, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. 2004. The automatic content extraction (ACE) programtasks, data, and evaluation. In *Proceedings of LREC*, volume 4, page 837840.

Finkel, Jenny R., Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, volume 43, page 363.

Hwa, Rebecca, Philip Resnik, Amy Weinberg, Clara Cabezas, and Okan Kolak. 2005. Bootstrapping parsers via syntactic projection across parallel texts. *Natural Language Engineering*, 11(03):311–325.

Joachims, Thorsten. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*, pages 137–142.

Kambhatla, Nanda. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 22, Barcelona, Spain. Association for Computational Linguistics.

Kim, Hansaem. 2006. Korean national corpus in the 21st century sejong project. In *Proceedings of the 13th NIJL International Symposium*, page 4954.

Klein, Dan and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 423–430, Sapporo, Japan. Association for Computational Linguistics.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, volume 1, pages 48–54.

Lee, Gary Geunbae, Jeongwon Cha, and Jong-Hyeok Lee. 2002. Syllable pattern-based unknown morpheme segmentation and estimation for hybrid part-of-speech tagging of korean. *Computational Linguistics*, 28(1):53–70.

Merlo, Paola, Suzanne Stevenson, Vivian Tsang, and Gianluca Allaria. 2002. A multilingual paradigm for automatic verb classification. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 207–214, Philadelphia, Pennsylvania. Association for Computational Linguistics.

Moschitti, Alessandro. 2006. Making tree kernels practical for natural language learning. In *Proceedings of EACL06*.

Och, Franz Josef and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March.

Pado, Sebastian and Mirella Lapata. 2009. Cross-lingual annotation projection of semantic roles. *Journal of Artificial Intelligence Research*, 36(1):307340.

Yarowsky, David and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies 2001*, pages 1–8, Pittsburgh, Pennsylvania. Association for Computational Linguistics.

Yarowsky, David, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the first international conference on Human language technology research*, pages 1–8, San Diego. Association for Computational Linguistics.

Zelenko, Dmitry, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106.

Zhang, Min, Jie Zhang, Jian Su, and Guodong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 825–832, Sydney, Australia. Association for Computational Linguistics.

Zhang, Zhu. 2004. Weakly-supervised relation classification for information extraction. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 581–588, Washington, D.C., USA. ACM.

Zhou, Guodong, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, page 434.

Zitouni, Imed and Radu Florian. 2008. Mention detection crossing the language barrier. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 600–609, Honolulu, Hawaii. Association for Computational Linguistics.

Evaluating N-gram based Evaluation Metrics for Automatic Keyphrase Extraction

Su Nam Kim, Timothy Baldwin
CSSE
University of Melbourne
sunamkim@gmail.com, tb@ldwin.net

Min-Yen Kan
School of Computing
National University of Singapore
kanmy@comp.nus.edu.sg

Abstract

This paper describes a feasibility study of n-gram-based evaluation metrics for automatic keyphrase extraction. To account for near-misses currently ignored by standard evaluation metrics, we adapt various evaluation metrics developed for machine translation and summarization, and also the R-precision evaluation metric from keyphrase evaluation. In evaluation, the R-precision metric is found to achieve the highest correlation with human annotations. We also provide evidence that the degree of semantic similarity varies with the location of the partially-matching component words.

1 Introduction

Keyphrases are noun phrases (NPs) that are representative of the main content of documents. Since they represent the key topics in documents, extracting good keyphrases benefits various natural language processing (NLP) applications such as summarization, information retrieval (IR) and question-answering (QA). Keyphrases can also be used in text summarization as semantic metadata (Barzilay and Elhadad, 1997; Lawrie et al., 2001; D'Avanzo and Magnini, 2005). In search engines, keyphrases supplement full-text indexing and assist users in creating good queries.

In the past, a large body of work on keyphrases has been carried out as an extraction task, utilizing three types of cohesion: (1) document cohesion, i.e. cohesion between documents and keyphrases (Frank et al., 1999; Witten et al., 1999; Matsuo and Ishizuka, 2004; Medelyan and Witten, 2006; Nguyen and Kan, 2007; Wan and Xiao, 2008); (2) keyphrase cohesion, i.e. cohesion among keyphrases (Turney, 2003); and (3) term cohesion, i.e. cohesion among terms in a keyphrase (Park et al., 2004).

Despite recent successes in keyphrase extraction (Frank et al., 1999; Turney, 2003; Park et al., 2004; Medelyan and Witten, 2006; Nguyen and Kan, 2007), current work is hampered by the inflexibility of standard metrics in evaluating different approaches. As seen in other fields, e.g. machine translation (MT) and multi-document summarization, the advent of standardized automatic evaluation metrics, combined with standardized datasets, has enabled easy comparison of systems and catalyzed the respective research areas. Traditionally, the evaluation of automatic keyphrase extraction has relied on the number of exact matches in author-assigned keyphrases and reader-assigned keyphrases. The main problem with this approach is that even small variants in the keyphrases are not given any credit. For example, given the gold-standard keyphrase *effective grid computing algorithm*, *grid computing algorithm* is a plausible keyphrase candidate and should be scored appropriately, rather than being naively evaluated as wrong. Additionally, author-assigned keyphrases and even reader-assigned keyphrases often have their own problems in this type of evaluation (Medelyan and Witten, 2006). For example, some keyphrases are often partly or wholly subsumed by other candidates or may not even occur in the document. Therefore, counting the exactly-matching candidates has been shown to be suboptimal (Jarmasz

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 572–580,
Beijing, August 2010

and Barriere, 2004).

Our goal in this paper is to evaluate the reliability of automatic evaluation metrics that better account for near-misses. Prior research based on semantic similarity (Jarmasz and Barriere, 2004; Mihalcea and Tarau, 2004; Medelyan and Witten, 2006) has taken the approach of using external resources such as large corpora, Wikipedia or manually-curated index words. While we acknowledge that these methods can help address the near-miss problem, they are impractical due to the effort required to compile the requisite resources for each individual evaluation exercise, and furthermore, the resources tend to be domain-specific. In order to design a cheap, practical and stable keyphrase evaluation metric, our aim is to properly account for these near-misses without reliance on costly external resources.

According to our analysis, the degree of semantic similarity of keyphrase candidates varies relative to the location of overlap. For example, the candidate *grid computing algorithm* has higher semantic similarity than *computing algorithm* with the gold-standard keyphrase *effective grid computing algorithm*. Also, *computing algorithm* is closer than *effective grid* to the same gold-standard keyphrase. From these observations, we infer that n-gram-based evaluation metrics can be applied to evaluating keyphrase extraction, but also that candidates with the same relative n-gram overlap are not necessarily equally good.

Our primary goal is to test the utility of n-gram based evaluation metrics to the task of keyphrase extraction evaluation. We test the following evaluation metrics: (1) evaluation metrics from MT and multi-document summarization (BLEU, NIST, METEOR and ROUGE); and (2) R-precision (Zesch and Gurevych, 2009), an n-gram-based evaluation metric developed specifically for keyphrase extraction evaluation which has yet to be evaluated against humans at the extraction task. Secondarily, we attempt to shed light on the bigger question of whether it is feasible to expect that n-gram-based metrics without access to external resources should be able to capture subtle semantic differences in keyphrase candidates. To this end, we experimentally verify the impact of lexical overlap of different types on keyphrase similarity, and use this as the basis for proposing a variant of R-precision.

In the next section, we present a brief primer on keyphrases. We then describe the MT and summarization evaluation metrics trialled in this research, along with R-precision, modified R-precision and a semantic similarity-based evaluation metric for keyphrase evaluation (Section 3). In Section 4, we discuss our gold-standard and candidate extraction method. We compare the evaluation metrics with human assigned scores for suitability in Section 5, before concluding the paper.

2 A Primer on Keyphrases

Keyphrases can be either simplex words (e.g. *query*, *discovery*, or *context-awareness*)[1] or larger N-bars/noun phrases (e.g. *intrusion detection*, *mobile ad-hoc network*, or *quality of service*). The majority of keyphrases are 1–4 words long (Paukkeri et al., 2008).

Keyphrases are normally composed of nouns and adjectives, but may occasionally contain adverbs (e.g. *dynamically allocated task*, or *partially observable Markov decision process*) or other parts of speech. They may also contain hyphens (e.g. *sensor-grouping* or *multi-agent system*) and apostrophes for possessives (e.g. *Bayes' theorem* or *agent's goal*).

Keyphrases can optionally incorporate PPs (e.g. *service quality* vs. *quality of service*). A variety of prepositions can be used (e.g. *incentive for cooperation, inequality in welfare, agent security via approximate policy*), although the genetive *of* is the most common.

Keyphrases can also be coordinated, either as simple nouns at the top level (e.g. *performance and scalability* or *group and partition*) or within more complex NPs or between N-bars (e.g. *history of past encounter and transitivity* or *task and resource allocation in agent system*).

When candidate phrases get too long, abbreviations also help to form valid keyphrases (e.g. *computer support collaborative work* vs. *CSCW*, or *partially observable Markov decision process* vs. *POMDP*).

[1] All examples in this section are taken from the data set outlined in Section 4.

3 Evaluation Metrics

There have been various evaluation metrics developed and validated for reliability in fields such as MT and summarization (Callison-Burch et al., 2009). While n-gram-based metrics don't capture systematic alternations in keyphrases, they do support partial match between keyphrase candidates and the reference keyphrases.

In this section, we first introduce a range of popular n-gram-based evaluation metrics from the MT and automatic summarization literature, which we naively apply to the task of keyphrase evaluation. We then present R-precision, an n-gram-based evaluation metric developed specifically for keyphrase evaluation, and propose a modified version of R-precision which weights n-grams according to their relative position in the keyphrase. Finally, we present a semantic similarity method.

3.1 Machine Translation and Summarization Evaluation Metrics

In this research, we experiment with four popular n-gram-based metrics from the MT and automatic summarization fields — BLEU, METEOR, NIST and ROUGE. The basic task performed by the respective evaluation metrics is empirical determination of *how good an approximation is string*$_1$ *of string*$_2$*?*, which is not far removed from the requirements of keyphrase evaluation. We briefly outline each of the methods below.

One subtle property of keyphrase evaluation is that there is no a priori preference for shorter keyphrases over longer keyphrases, unlike MT where shorter strings tend to be preferred. Hence, we use the longer NP as reference and the shorter NP as a translation, to avoid the length penalty in most MT metrics.[2]

BLEU (Papineni et al., 2002) is an evaluation metric for measuring the relative similarity between a candidate translation and a set of reference translations, based on n-gram composition. It calculates the number of overlapping n-grams between the candidate translation and the set of reference translations. In order to avoid having very short translations receive artificially high scores, BLEU adds a brevity penalty to the scoring equation.

METEOR (Agarwal and Lavie, 2008) is similar to BLEU, in that it measures string-level similarity between the reference and candidate translations. The difference is that it allows for more match flexibility, including stem variation and WordNet synonymy. The basic metric is based on the number of mapped unigrams found between the two strings, the total number of unigrams in the translation, and the total number of unigrams in the reference.

NIST (Martin and Przybocki, 1999) is once again similar to BLEU, but integrates a proportional difference in the co-occurrences for all n-grams while weighting more heavily n-grams that occur less frequently, according to their information value.

ROUGE (Lin and Hovy, 2003) — and its variants including ROUGE-N and ROUGE-L — is similarly based on n-gram overlap between the candidate and reference summaries. For example, ROUGE-N is based on co-occurrence statistics, using higher-order n-grams ($n > 1$) to estimate the fluency of summaries. ROUGE-L uses longest common subsequence (LCS)-based statistics, based on the assumption that the longer the substring overlap between the two strings, the greater the similar Saggion et al. (2002). ROUGE-W is a weighted LCS-based statistic that prioritizes consecutive LCSes. In this research, we experiment exclusively with the basic ROUGE metric, and unigrams (i.e. ROUGE-1).

3.2 R-precision

In order to analyze near-misses in keyphrase extraction evaluation, Zesch and Gurevych (2009) proposed R-precision, an n-gram-based evaluation metric for keyphrase evaluation.[3] R-precision contrasts with the majority of previous work on keyphrase extraction evaluation, which has used semantic similarity based on external resources

[2]While we don't present the numbers in this paper, the results were lower for the MT evaluation metrics without this reordering of the reference and candidate keyphrases.

[3]Zesch and Gurevych's R-precision has nothing to do with the information retrieval evaluation metric of the same name, where P@N is calculated for N equal to the number of relevant documents.

(Jarmasz and Barriere, 2004; Mihalcea and Tarau, 2004; Medelyan and Witten, 2006). As our interest is in fully automated evaluation metrics which don't require external resources and are domain independent (for maximal reproducibility of results), we experiment only with R-precision in this paper.

R-precision is based on the number of overlapping words between a keyphrase and a candidate, as well as the length of each. The metric differentiates three types of near-misses: *INCLUDE*, *PARTOF* and *MORPH*. The first two types are based on an n-gram approach, while the third relies on lexical variation. As we use stemming, in line with the majority of previous work on keyphrase extraction evaluation, we focus exclusively on the first two cases, namely *INCLUDE*, and *PARTOF*. The final score returned by R-precision is:

$$\frac{\text{number of overlapping word(s)}}{\text{length of keyphrase/candidate}}$$

where the denominator is the longer of the keyphrase and candidate.

Zesch and Gurevych (2009) evaluated R-precision over three corpora (Inspec, DUC and SP) based on 566 non-exact matching candidates. In order to evaluate the human agreement, they hired 4 human annotators to rate the near-miss candidates, and reported agreements of 80% and 44% for the INCLUDE and PARTOF types, respectively. They did not, however, perform holistic evaluation with human scores to verify its reliability in full system evaluation. This is one of our contributions in this paper.

3.3 Modified R-precision

In this section, we describe a modification to R-precision which assigns different weights for component words based on their position in the keyphrase (unlike R-precision which assigns the same score for each matching component word). The head noun generally encodes the core semantics of the keyphrase, and as a very rough heuristic, the further a word is from the head noun, the less semantic import on the keyphrase it has. As such, modified R-precision assigns a score to each component word relative to its position as $CW = \frac{1}{N-i+1}$ where N is the number of component words in the keyphrase and i is the position of the component word in the keyphrase (1 = leftmost word).

For example, *AB* and *BC* from *ABC* would be scored as $\frac{\frac{1}{3}+\frac{1}{2}}{\frac{1}{3}+\frac{1}{2}+\frac{1}{1}} = \frac{5}{11}$ and $\frac{\frac{1}{2}+\frac{1}{1}}{\frac{1}{3}+\frac{1}{2}+\frac{1}{1}} = \frac{9}{11}$, respectively. Thus, with the keyphrase *effective grid computing algorithm* and candidates *effective grid*, *grid computing* and *computing algorithm*, modified R-precision assigns different scores for each candidate (*computing algorithm* > *grid computing* > *effective grid*). In contrast, the original R-precision assigns the same score to all candidates.

3.4 Semantic Similarity

In Jarmasz and Barriere (2004) and Mihalcea and Tarau (2004), the authors used a large data set to compute the semantic similarity of two NPs to assign partial credits for semantically similar candidate keyphrases. To simulate these methods, we adopted the distributional semantic similarity using web documents. That is, we computed the similarity between a keyphrase and its substring by cosine measure over collected the snippets from `Yahoo! BOSS`.[4] We use the computed similarity as our score for near-misses.

4 Data

4.1 Data Collection

We constructed a keyphrase extraction dataset using papers across 4 different categories[5] of the ACM Digital Library.[6] In addition to author-assigned keyphrases provided as part of the ACM Digital Library, we generated reader-assigned keyphrases by assigning 250 students 5 papers each, a list of candidate keyphrases (see below for details), and standardized instructions on how to assign keyphrases. It took them an average of 15 minutes to annotate each paper. This is the same

[4] `http://developer.yahoo.com/search/boss/`

[5] C2.4 (Distributed Systems), H3.3 (Information Search and Retrieval), I2.11 (Distributed Artificial Intelligence – Multiagent Systems) and J4 (Social and Behavioral Sciences – Economics).

[6] `http://portal.acm.org/dl.cfm`

	Author	Reader	Total
Total	1298/1305	3110/3221	3816/3962
NPs	937	2537	3027
Average	3.85/4.01	12.44/12.88	15.26/15.85
Found	769	2509	2864

Table 1: Details of the keyphrase dataset

(**Rule1**) NBAR = (`NN*`|`JJ*`)*(`NN*`)
e.g. *complexity, effective algorithm, distributed web-service discovery architecture*
(**Rule2**) NBAR `IN` NBAR
e.g. *quality of service, sensitivity of VOIP traffic, simplified instantiation of zebroid*

Table 2: Regular expressions for candidate selection

document collection and set of keyphrase annotations as was used in the SemEval 2010 keyphrase extraction task (Kim et al., 2010).

Table 1 shows the details of the final dataset. The numbers after the slashes indicate the number of keyphrases after including alternate keyphrases based on *of*-PPs. Despite the reliability of author-assigned keyphrases discussed in Medelyan and Witten (2006), many author-assigned keyphrases and some reader-assigned keyphrases are not found verbatim in the source documents because: (1) many of them are substrings of the candidates or vice versa (about 75% of the total keyphrases are found in the documents); and (2) our candidate selection method does not extract keyphrases in forms such as coordinated NPs or adverbial phrases.

4.2 Candidate Selection

During preprocessing, we first converted the PDF versions of the papers into text using `pdftotext`. We then lemmatized and POS tagged all words using `morpha` and the `Lingua` POS tagger. Next, we applied the regular expressions in Table 2 to extract candidates, based on Nguyen and Kan (2007). Finally, we selected candidates in terms of their frequency: simplex words with frequency ≥ 2 and NPs with frequency ≥ 1. We observed that for reader-assigned keyphrases, NPs were often selected regardless of their frequency in the source document. In addition, we allowed variation in the possessive form, noun number and abbreviations.

Rule1 detects simplex nouns or N-bars as candidates. *Rule2* extracts N-bars with post-modifying PPs. In Nguyen and Kan (2007), *Rule2* was not used to additionally extract N-bars inside modifying PPs. For example, our rules extract not only *performance of grid computing* as a candidate, but also *grid computing*. However, we did not extend the candidate selection rules to cover NPs including adverbs (e.g. *partially-observable Markov decision process*) or conjunctions (e.g. *behavioral evolution and extrapolation*), as they are rare.

4.3 Human Assigned Score

We hired four graduate students working in NLP to assign human scores to substrings in the gold-standard data. The scores are between 0 and 4 (0 means no semantic overlap between a NP and its substring, while 4 means semantically indistinguishable).

We broke down the candidate–keyphrases pairs into subtypes, based on where the overlap occurs relative to the keyphrase (e.g. *ABCD*): (1) *Head*: the candidate contains the head noun of the keyphrase (e.g. *CD*); (2) *First*: the candidate contains the first word of the keyphrase (e.g. *AB*); and (3) *Middle*: the candidate overlaps with the keyphrase, but contains neither its first word nor its head word (e.g. *BC*). The average human scores are 1.94 and 2.11 for *First* and *Head*, respectively, when the candidate is shorter, while they are 2.00, 1.89 and 2.15 for *First*, *Middle*, and *Head*, respectively when the candidate is longer. Note that we did not have *Middle* instances with candidates as the shorter string. The scores are slightly higher for the keyphrases as substrings than for the candidates as substrings.

5 Correlation

To check the feasibility of metrics for keyphrase evaluation, we checked the Spearman rank correlation between the machine-generated score and the human-assigned score for each keyphrase–candidate pairing.

As the percentage of annotators who agree on the exact score is low (i.e. 2 subjects agree ex-

		Human	R-precision Orig	R-precision Mod	BLEU	METEOR	NIST	ROUGE	Semantic Similarity
Average	All	.4506	.4763	.2840	.3250	.3246	.3366	.3246	.2116
	$L \leq 4$	.4510	.5264	.2806	.3242	.3238	.3369	.3240	.2050
	$L \leq 3$	.4551	.4834	.2893	.3439	.3437	.3584	.3437	.1980
Majority	All	.4603	.4763	.3438	.3407	.3403	.3514	.3404	.2224
	$L \leq 4$	.4604	.5264	.3434	.3423	.3421	.3547	.3422	.2168
	$L \leq 3$	.4638	.4838	.3547	.3679	.3675	.3820	.3676	.2123

Table 3: Rank correlation between humans and the different evaluation metrics, based on the human average (top half) and majority (bottom half)

		Human	R-precision Orig	R-precision Mod	BLEU	METEOR	NIST	ROUGE
LOCATION	First	**.5508**	**.5032**	**.5033**	.3844	.3844	.4057	.3844
	Middle	**.5329**	**.5741**	**.5988**	**.4669**	**.4669**	.4055	**.4669**
	Head	**.3783**	**.4838**	**.4838**	.3865	.3860	.3780	.3864
COMPLEXITY	Simple	.4452	**.4715**	.2790	.3653	.3445	.3527	.3445
	PP	**.4771**	**.4814**	.1484	.3367	.3122	.3443	.3123
	CC	.3645	.3810	.3140	.3748	.3446	.3384	.3748
POS	AdjN	**.4616**	**.4844**	.3507	.3147	.3132	.3115	.3133
	NN	.4467	**.4586**	.2581	.3321	.3321	.3488	.3322

Table 4: Rank correlation between human average judgments and n-gram-based metrics

actly on 55%-70% of instances, 3 subjects agree exactly on 25%-35% of instances), we require a method for combining the annotations. We experiment with two combination methods: majority and average. The majority is simply the label with the majority of annotations associated with it; in the case of a tie, we break the tie by selecting that annotation which is closest to the median. The average is simply the average score across all annotators.

5.1 Overall Correlation with Human Scores

Table 3 presents the correlations between the human scores (acting as an upper bound for the task), as well as those between human scores with machine-generated scores. We first present the overall results, then results over the subset of keyphrases of length 4 words or less, and also 3 words or less. We present the results for the annotator average and majority in top and bottom half, respectively, of the table.

To compute the correlation between the human annotators, we used leave-one-out cross-validation, holding out one annotator, and comparing them to the combination of the remaining annotators (using either the majority or average method to combine the remaining annotations). This was repeated across all annotators, and the Spearman's ρ was averaged across the annotators.

Overall, we found that R-precision achieved the highest correlation with humans, above the inter-annotator correlation in all instances. That is, based on the evaluation methodology employed, it is performing slightly above the average level of a single annotator. The relatively low inter-annotator correlation is, no doubt, due to the difficulty of the task, as all of our near-misses have 2 or more terms, and the annotators have to make very fine-grained, and ultimately subjective, decisions about the true quality of the candidate.

Comparing the n-gram-based methods with the semantic similarity-based method, the n-gram-based metrics achieved higher correlations across the board, with BLEU, METEOR, NIST and ROUGE all performing remarkably consistently, but well

		Human	R-precision Orig	Mod	BLEU	METEOR	NIST	ROUGE
LOCATION	First	**.5642**	**.5162**	**.5163**	.4032	.4032	.4297	.4032
	Middle	**.5510**	**.4991**	**.5320**	.4175	.4175	.3653	.4175
	Head	.4147	**.5073**	**.5074**	.4156	.4153	.4042	.4156
COMPLEXITY	Simple	.4580	**.4869**	.3394	.3653	.3651	.3715	.3651
	PP	**.4715**	**.5068**	.3724	.3367	.3367	.3652	.3367
	CC	**.5777**	**.5513**	.3841	**.5745**	**.5571**	**.5600**	**.5745**
POS	AdjN	.4501	**.4861**	.3968	.3266	.3251	.3246	.3252
	NN	**.4631**	**.4733**	.3244	.3499	.3499	.3648	.3500

Table 5: Rank correlation between human majority and n-gram-based metrics

below the level of R-precision. Due to the markedly lower performance of the semantic similarity-based method, we do not consider it for the remainder of our experiments. A general finding was that as the length of the keyphrase (L) got longer, the correlation tended to be higher across all n-gram-based metrics.

One disappointment at this stage is that the results for modified R-precision are well below those of the original, especially over the average of the human annotators.

5.2 Correlation with Different NP Subtypes

To get a clearer sense of how the different evaluation metrics are performing, we broke down the keyphrases according to three syntactic subclassifications: (1) the location of overlap (see Section 4.3); (2) the complexity of the NP (does the keyphrase contain a preposition [PP], a conjunction [CC] or neither a preposition nor a conjunction [Simple]?); and (3) the word class sequence of the keyphrase (is the keyphrase an NN [NN] or an AdjN sequence [AdjN]?). We present the results in Tables 4 and Table 4 for the human average and majority, respectively, presenting results in **boldface** when the correlation for a given method is higher than for that same method in our holistic evaluation in Table 3 (i.e. .4506 and .4603, for the average and majority human scores, respectively).

All methods, including inter-annotator correlation, improve in raw numbers over the subsets of the data based on overlap location, indicating that the data was partitioned into more internally-consistent subsets. Encouragingly, modified R-precision equalled or bettered the performance of the original R-precision over each subset of the data based on overlap location. Where modified R-precision appears to fall down most noticeably is over keyphrases including prepositions, as our assumption about the semantic import based on linear ordering clearly breaks down in the face of post-modifying PPs. It is also telling that it does worse over noun–noun sequences than adjective–noun sequences. In being agnostic to the effects of syntax, the original R-precision appears to benefit overall. Another interesting effect is that the performance of BLEU, METEOR and ROUGE is notably better over candidates which match with non-initial and non-final words in the keyphrase.

We conclude from this analysis that keyphrase scoring should be sensitive to overlap location. Furthermore, our study also shows that n-gram-based MT and summarization metrics are surprisingly adept at capturing partial matches in keyphrases, despite them being much shorter than the strings they are standardly applied to. More compellingly, we found that R-precision is the best overall performer, and that it matches the performance of our human annotators across the board. This is the first research to establish this fact. Our findings for modified R-precision were more sobering, but its location sensitivity was shown to improve over R-precision for instances of overlap in the middle or with the head of the keyphrase.

6 Conclusion

In this work, we have shown that preexisting n-gram-based evaluation metrics from MT, summarization and keyphrase extraction evaluation are able to handle the effects of near-misses, and that R-precision performs at or above the average level of a human annotator. We have also shown that a semantic similarity-based method which uses web data to model distributional similarity performed below the level of all of the n-gram-based methods, despite them requiring no external resources (web or otherwise). We proposed a modification to R-precision based on the location of match, but found that while it could achieve better performance over certain classes of keyphrases, its net effect was to drag the performance of R-precision down. Other methods were found to be remarkably consistent across different subtypes of keyphrase.

Acknowledgements

Many thanks to the anonymous reviewers for their insightful comments. We wish to acknowledge the generous funding from National Research Foundation grant R 252-000-279-325 in supporting Min-Yen Kan's work.

References

Abhaya Agrwal and Alon Lavie. METEOR, M-BLEU and M-TER: Evaluation Metrics for High-Correlation with Human Rankings of Machine Translation Output. In *Proceedings of ACL Workshop on Statistical Machine Translation*. 2008.

Ken Barker and Nadia Cornacchia. Using noun phrase heads to extract document keyphrases. In *Proceedings of BCCSCSI : Advances in Artificial Intelligence*. 2000, pp.96–103.

Regina Barzilay and Michael Elhadad. Using lexical chains for text summarization. In *Proceedings of ACL/EACL Workshop on Intelligent Scalable Text Summarization*. 1997, pp. 10–17.

Chris Callison-Burch, Philipp Koehn, Christof Monz and Josh Schroeder. Proceedings of 4th Workshop on Statistical Machine Translation. 2009.

Ernesto D'Avanzo and Bernado Magnini. A Keyphrase-Based Approach to Summarization: the LAKE System at DUC-2005. In *Proceedings of DUC*. 2005.

Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin and Craig G. Nevill-Manning. Domain Specific Keyphrase Extraction. In *Proceedings of IJCAI*. 1999, pp.668–673.

Mario Jarmasz and Caroline Barriere. Using semantic similarity over Tera-byte corpus, compute the performance of keyphrase extraction. In *Proceedings of CLINE*. 2004.

Su Nam Kim, Olena Medelyan, Min-Yen Kan and Timothy Baldwin. SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles. In *Proceedings of SemEval-2: Evaluation Exercises on Semantic Evaluation*. to appear.

Dawn Lawrie, W. Bruce Croft and Arnold Rosenberg. Finding Topic Words for Hierarchical Summarization. In *Proceedings of SIGIR*. 2001, pp. 349–357.

Chin-Yew Lin and Edward H. Hovy. Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In *In Proceedings of HLT-NAACL*. 2003.

Alvin Martin and Mark Przybocki. The 1999 NIST Speaker Recognition Evaluation, Using Summed Two-Channel Telephone Data for Speaker Detection and Speaker Tracking. In *Proceedings of EuroSpeech*. 1999.

Yutaka Matsuo and Mitsuru Ishizuka. Keyword Extraction from a Single Document using Word Co-occurrence Statistical Information. *International Journal on Artificial Intelligence Tools*. 2004, 13(1), pp. 157–169.

Olena Medelyan and Ian Witten. Thesaurus based automatic keyphrase indexing. In *Proceedings of ACM/IEED-CS JCDL*. 2006, pp. 296–297.

Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Texts. In *Proceedings of EMNLP 2004*. 2004, pp. 404–411.

Guido Minnen, John Carroll and Darren Pearce. Applied morphological processing of English. *NLE*. 2001, 7(3), pp. 207–223.

Thuy Dung Nguyen and Min-Yen Kan. Key phrase Extraction in Scientific Publications. In *Proceeding of ICADL*. 2007, pp. 317–326.

Sebastian Padó, Michel Galley, Dan Jurafsky and Christopher D. Manning. Textual Entailment Features for Machine Translation Evaluation. In *Proceedings of ACL Workshop on Statistical Machine Translation*. 2009, pp. 37–41.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of ACL*. 2001, pp. 311–318.

Youngja Park, Roy J. Byrd and Branimir Boguraev. Automatic Glossary Extraction Beyond Terminology Identification. In *Proceedings of COLING*. 2004, pp. 48–55.

Mari-Sanna Paukkeri, Ilari T. Nieminen, Matti Polla and Timo Honkela. A Language-Independent Approach to Keyphrase Extraction and Evaluation. In *Proceedings of COLING*. 2008, pp. 83–86.

Horacio Saggion, Dragomir Radev, Simon Teufel, Wai Lam and Stephanie Strassel. Meta-evaluation of Summaries in a Cross-lingual Environment using Content-based Metrics. In *Proceedings of COLING*. 2002, pp. 1–7.

Peter Turney. Coherent keyphrase extraction via Web mining. In *Proceedings of IJCAI*. 2003, pp. 434–439.

Xiaojun Wan and Jianguo Xiao. CollabRank: towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of COLING*. 2008, pp. 969–976.

Ian Witten, Gordon Paynter, Eibe Frank, Car Gutwin and Craig Nevill-Manning. KEA:Practical Automatic Key phrase Extraction. In *Proceedings of ACM conference on Digital libraries*. 1999, pp. 254–256.

Torsten Zesch and Iryna Gurevych. Approximate Matching for Evaluating Keyphrase Extraction. *International Conference on Recent Advances in Natural Language Processing*. 2009.

Improving the Quality of Text Understanding by Delaying Ambiguity Resolution

Doo Soon Kim
Dept. of Computer Science
University of Texas
onue5@cs.utexas.edu

Ken Barker
Dept. of Computer Science
University of Texas
kbarker@cs.utexas.edu

Bruce Porter
Dept. of Computer Science
University of Texas
porter@cs.utexas.edu

Abstract

Text Understanding systems often commit to a *single best* interpretation of a sentence before analyzing subsequent text. This interpretation is chosen by resolving ambiguous alternatives to the one with the highest confidence, given the context available at the time of commitment. Subsequent text, however, may contain information that changes the confidence of alternatives. This may especially be the case with multiple redundant texts on the same topic. Ideally, systems would delay choosing among ambiguous alternatives until more text has been read.

One solution is to maintain multiple candidate interpretations of each sentence until the system acquires disambiguating evidence. Unfortunately, the number of alternatives explodes quickly. In this paper, we propose a *packed graphical (PG) representation* that can efficiently represent a large number of alternative interpretations along with dependencies among them. We also present an algorithm for combining multiple PG representations to help resolve ambiguity and prune alternatives when the time comes to commit to a single interpretation.

Our controlled experiments show that by delaying ambiguity resolution until multiple texts have been read, our prototype's accuracy is higher than when committing to interpretations sentence-by-sentence.

1 Introduction

A typical text understanding system confronts ambiguity while parsing, mapping words to concepts and formal relations, resolving co-references, and integrating knowledge derived from separate sentences or texts. The system discards many candidate interpretations to avoid combinatorial explosion. Commonly, after reading each sentence, a system will commit to its top ranked interpretation of the sentence before reading the next.

If a text understanding system could postpone committing to an interpretation without being swamped by a combinatorial explosion of alternatives, its accuracy would almost surely improve. This intuition follows from the observation that text is redundant in at least two ways. First, within a single coherent text (about the same entities and events), each sentence informs the interpretation of its neighbors. Second, within a corpus of texts on the same topic, the same information is expressed in different surface forms, ambiguous in different ways. Related fields, such as Information Extraction, exploit textual redundancy to good effect, and perhaps text understanding can as well.

One approach is for the text understanding system to maintain multiple complete candidate interpretations. After reading each sentence, for example, the system would retain a beam of the n-best interpretations of the sentence. While this approach avoids a combinatorial explosion (for reasonable values of n), several problems remain. First, because the beam width is limited, the system may still discard correct interpretations before benefiting from the extra context from related text. Second, enumeration of the candidate interpreta-

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 581–589,
Beijing, August 2010

tions does not represent the dependencies among them. For example, there may be multiple candidate word senses and semantic roles for a given sentence, but sense alternatives might be dependent on role selection (and vice-versa). The set of reasonable interpretations may be a subset of all combinations. Finally, maintaining distinct interpretations does not contribute to addressing the problem of combining evidence to narrow down alternatives and ultimately select a single best interpretation of a text.

This paper addresses these three problems. We propose an approach that postpones committing to an interpretation of a text by representing ambiguities and the dependencies among them. There may still be combinatorial growth in the set of alternative interpretations, but they are represented only intensionally, using a packed representation, which maintains alternatives while avoiding enumerating them. We also propose an algorithm for updating and pruning the packed representation as more sentences and texts are read.

We evaluate our approach by comparing two reading systems: a baseline system that commits to its best interpretation after each sentence, and our prototype system that uses a packed representation to maintain all interpretations until further reading enables it to prune. For this initial proof of concept, we use a small corpus of redundant texts. The results indicate that our approach improves the quality of text interpretation by preventing aggressive pruning while avoiding combinatorial explosion.

In the following sections, we first describe our target semantic representation of the interpretation of sentences. We then present the details of our *packed graphical representation (PG representation)* and our algorithm to resolve ambiguities in the PG representations as disambiguating evidence from subsequent text accrues. We describe the architecture of a prototype that produces PG representations for text and implements the disambiguating algorithm. Finally, we present the results from controlled experiments designed to compare the accuracy of the prototype to a baseline system that prunes more aggressively.

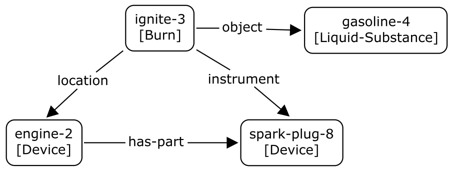


Figure 1: The target semantic graph representation for S1

2 Target semantic representation

Our target representation is a semantic graph in which nodes are words and the ontological types to which they map. Edges are semantic relations corresponding either to function words or syntactic relations in the sentence's parse.

Fig. 1 shows the target semantic representation for the following simple sentence:

S1: *An engine ignites gasoline with its spark plug.*

3 PG representation

Alternative semantic interpretations for a sentence can be captured with a single PG representation with ambiguities represented as local alternatives. Because candidate representations are often structurally similar, a PG representation can significantly compress the representation of alternatives.

Fig. 2 shows the PG representation of alternate interpretations of S1 (PG1). The different types of ambiguity captured by the PG representation are as follows.

3.1 Word-Type ambiguity

In PG1, the node engine-2a corresponds to the word "engine" in S1. Its annotation [LIVING-ENTITY .3 | DEVICE .7] captures the mapping to either LIVING-ENTITY (probability 0.3) or DEVICE (probability 0.7). The PG representation does not presume a particular uncer-

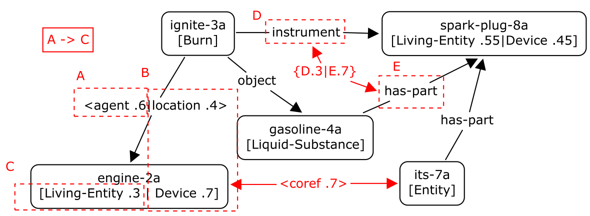


Figure 2: The PG representation for S1 (PG1)

tainty formalism. Any formalism, (Dempster-Shafer theory (Pearl, 1988), Markov Logic Networks (Richardson and Domingos, 2006), etc.) could be used.

3.2 Semantic Relation ambiguity

In PG1, the edge label <agent .6 | location .4> from ignite-3a to engine-2a says that the engine is either *agent* or *location* of the ignition.

3.3 Structural ambiguity

In PG1, edges D and E are alternatives corresponding to the different prepositional phrase attachments for "with its spark plug" (to ignite-3a or gasoline-4a). The annotation {D .3 | E .7} says that the choices are mutually exclusive with probabilities of 0.3 and 0.7.

3.4 Co-reference ambiguity

Co-reference of nodes in a PG representation is captured using a "co-reference" edge. In PG1, the edge labeled <coref .7> represents the probability that engine-2a and its-7a are co-referent.

In addition to storing ambiguities explicitly, the PG representation also captures dependencies among alternatives.

3.5 Simple dependency

The existence of one element in the graph depends on the existence of another element. If subsequent evidence suggests that an element is incorrect, its dependents should be pruned. For example, the dependency A $\rightarrow$ C, means that if LIVING-ENTITY is ultimately rejected as the type for engine-2a, the agent relation should be pruned.

3.6 Mutual dependency

Elements of a mutual dependency set are mutually confirming. Evidence confirming or rejecting an element also confirms or rejects other elements in the set. In the example, the box labeled B says that (engine-2a type DEVICE) and (ignite-3a location engine-2a) should both be confirmed or pruned when either of them is confirmed or pruned.

Formally, the PG representation is a structure consisting of (a) *semantic triples* – e.g., (ignite-3a type BURN), (b) *macros* – e.g., the symbol A refers to (ignite-3a agent engine-2a), and (c) *constraints* – e.g., A depends on C.

4 Combining PG representations

Maintaining ambiguity within a PG representation allows us to delay commitment to an interpretation until disambiguating evidence appears. For any text fragment that results in a PG representation (PGa) containing ambiguity, there may exist other text fragments that are partly redundant, but result in a less ambiguous (or differently ambiguous) representation (PGb). PGb can be used to adjust confidences in PGa. Enough such evidence allows us to prune unlikely interpretations, ultimately disambiguating the original representation.

For example, sentence S3 does not have sufficient context to disambiguate between the MOTOR sense of "engine" and the VEHICLE sense (as in *locomotive*).

S3: *General Electric announced plans this week for their much anticipated new engine.*

The PG3 representation for S3 (PG3) would maintain the ambiguous representation (with confidences for each sense based on prior probabilities, for example). On subsequently encountering sentence S4, a Lesk-based word sense disambiguation module (as in our prototype) would produce a PG4 with a strong preference for the locomotive sense of "engine", given the more specific context of S4.

S4: *The announcement comes to the relief of many in the railway industry looking to replace the engines in their aging locomotive fleets.*

To use PG4 to help disambiguate PG3, we need to align PG3 and PG4 semantically and merge their conflict sets. (In the simple example, the conflict sets for the word "engine" might be [MOTOR .5 | VEHICLE .5] in PG3 and [MOTOR .2 | VEHICLE .8] in PG4).

Algorithm 1 describes how two PG representations can be combined to help resolve their ambiguities. The algorithm identifies their isomorphic subgraphs (redundant portions of the interpretations) and uses the information to disambiguate their ambiguities. For illustration, we will step through Algorithm 1, merging PG1 (Fig. 2) with

Algorithm 1 Disambiguating PG representations

Input : PG1, PG2

Output: new PG representation

1. *Identify semantically aligned parts between PG1 and PG2.* Use graph matching to identify alignments (redundant portions) between PG1 and PG2: align nodes with the same base word or with taxonomically related types; from the node alignments, align identical types as type alignments; align relations if the relations are the same and their head and tail nodes have been aligned.
2. *Use alignments to disambiguate PG1 and PG2.* With the available information (the confidence scores and the constraints in PG1 and PG2 and the alignments between them), use joint inference to calculate the confidence score of each candidate interpretation. If the confidence score of one interpretation becomes much higher than competing ones, the interpretation is chosen while the others are discarded.
3. *Combine the disambiguated PG1 and PG2 into one PG representation using the alignments identified in the first step.*

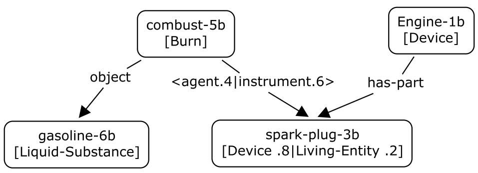


Figure 3: PG representation for S2, *"The engine's spark plug combusts gasoline."*

PG2 (Fig. 3).

1. The graph matcher identifies alignments between PG1 and PG2. Type alignments include (engine-2a[DEVICE], Engine-1b[DEVICE]), (spark-plug-8a[LIVING-ENTITY], spark-plug-3b[LIVING-ENTITY]). Relation alignments include ((combust-5b instrument spark-plug-3b), (ignite-3 instrument spark-plug-8)), ((ignite-3a instrument spark-plug-8a) (combust-5b instrument spark-plug-3b)).

2. In this example, when two interpretations are aligned, we simply add their confidence scores. (We are currently incorporating *Alchemy* (Richardson and Domingos, 2006) in the prototype system to do the joint inference). For example, aligning engine-2a with Engine-1b results in a score of 1.7 for DEVICE (1 + .7). The confidence score of LIVING-ENTITY in engine-2a is unchanged at .3. Since the resulting score for DEVICE is much higher than [1] the score for LIVING-ENTITY, LIVING-ENTITY is discarded. Deleting LIVING-ENTITY causes deletion of the *agent* edge between ignite-3a and engine-2a due to the dependency constraint A $\rightarrow$ C.

3. The disambiguated PG1 and PG2 are merged into a single PG representation (PG1+2) based on the alignments. Any remaining ambiguity persists in PG1+2, possibly to be resolved with another sentence.

5 Prototype system

5.1 Parser

Our prototype system uses the Stanford Parser (Klein and Manning, 2003). To capture structural ambiguity for our experiments, we manually edited the parser output by adding corrections as alternatives wherever the parse tree was incorrect. This gave a syntactic PG representation with both incorrect and correct alternatives. We gave the original, incorrect alternatives high confidence scores and the added, correct alternatives low scores, simulating a parser pruning correct interpretations in favor of incorrect ones with higher confidence scores. The syntactic PG for S1 is shown in Fig. 4. We have recently designed a modification to the Stanford Parser to make it produce syntactic PG representations natively, based on the complete chart built during parsing.

5.2 Semantic Interpreter

The semantic interpreter assigns types to nodes in the syntactic PG representation and semantic relations to the edges.

Type ambiguity. Types and confidence scores are assigned to words using SenseRelate (Patwardhan et al., 2005), WSD software based on the

[1]In our prototype, we set the pruning threshold at $\frac{1}{3}\times$the score of the top-scored interpretation.

Lesk Algorithm (Lesk, 1986). Assigned senses are then mapped to our *Component Library* ontology (Barker et al., 2001) using its built-in WordNet mappings.

Relational ambiguity. Semantic relations are assigned to the dependency relations in the syntactic PG representation according to semantic interpretation rules. Most rules consider the head and tail types as well as the dependency relation, but do not produce confidence scores. Our prototype scores candidates equally. We plan to incorporate a more sophisticated scoring method such as (Punyakanok et al., 2005).

Structural ambiguity. Parse ambiguities (such as PA vs. PB in Fig. 4) are converted directly to structural ambiguity representations (D vs. E in Fig. 2) in the semantic PG representation.

Simple Dependency. A dependency is installed between a type t for word w and a semantic relation r when (1) r is produced by a rule based on t and (2) r is dependent on no other candidate type for w. In Fig. 2, a dependency relation is installed from A to C, because (1) LIVING-ENTITY in engine-2a was used in the rule assigning *agent* between ignite-3a and engine-2a and (2) the assignment of *agent* is not dependent on DEVICE, the other candidate type of engine-2a.

Mutual dependency. If multiple interpretations depend on one another, a mutual dependency set is created to include them.

5.3 PG Merger

The PG Merger implements Algorithm 1 to combine PG representations. The PG representation

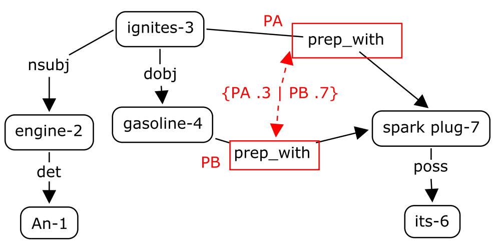


Figure 4: Syntactic PG representation for S1, capturing the PP-attachment ambiguity of "with its spark plug".

Original Text Hearts pump blood through the body. Blood carries oxygen to organs throughout the body. Blood leaves the heart, then goes to the lungs where it is oxygenated. The oxygen given to the blood by the lungs is then burned by organs throughout the body. Eventually the blood returns to the heart, depleted of oxygen.
Paraphrase The heart begins to pump blood into the body. The blood first travels to the lungs, where it picks up oxygen. The blood will then be deposited into the organs, which burn the oxygen. The blood will then return to the heart, where it will be lacking oxygen, and start over again.

Figure 5: The original text and a paraphrase

for each sentence is merged with the cumulative PG from previous sentences. The global PG representation integrates sentence-level PG representations to the extent that they align semantically. In the worst case (completely unrelated sentences), the global PG representation would simply be the union of individual PG representations. The extent to which the global PG is more coherent reflects redundancy and semantic overlap in the sentences.

6 Experiment 1

We first wanted to evaluate our hypothesis that Algorithm 1 can improve interpretation accuracy over multiple redundant texts. We manually generated ten redundant texts by having volunteers rewrite a short, tutorial text, using Amazon Turk (http://mturk.com) [2] The volunteers had no knowledge of the purpose of the task, and were asked to rewrite the text using "different" language. Fig. 5 shows the original text and one volunteer's rewrite. The total number of sentences over the ten texts was 37. Average sentence length was 14.5 words.

6.1 Evaluation Procedure

We ran two systems over the ten texts. The baseline system commits to the highest scoring consistent interpretation after each sentence. The prototype system produces an ambiguity-preserving

[2] We ultimately envision a system whose task is to develop a model of a particular topic by interpreting multiple texts. Such a system might be given a cluster of documents or use its own information retrieval to find similar documents given a tutorial text.

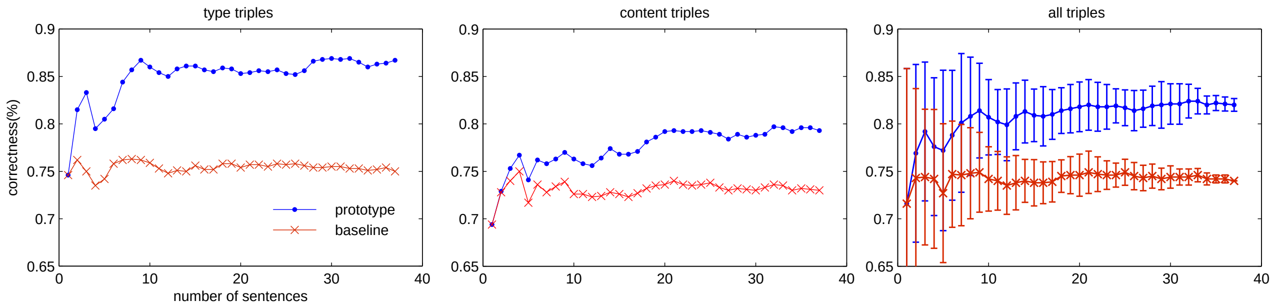


Figure 6: Correctness scores for the prototype vs. baseline system on (a) type triples (word sense assignment), (b) content triples (semantic relations) and (c) all triples (with standard deviation).

PG representation. For each sentence, the prototype's PG Merger merges the PG of the sentence with the merged PG of the previous sentences. After N sentences (varying N from 1..37), the system is forced to commit to the highest scoring consistent interpretation in the merged PG. For N=1 (commit after the first sentence), both the baseline and prototype produce the same result. For N=2, the baseline produces the union of the highest scoring interpretations for each of the first two sentences. The prototype produces a merged PG for the first two sentences and then prunes to the highest scoring alternatives.

At each value of N, we measured the correctness of the interpretations (the percentage of correct semantic triples) for each system by comparing the committed triples against human-generated gold standard triples.

We repeated the experiment ten times with different random orderings of the 37 sentences, averaging the results.

6.2 Evaluation result

Fig. 6 shows that both type assignment and semantic relation assignment by the prototype improve as the system reads more sentences. This result confirms our hypothesis that delaying commitment to an interpretation resolves ambiguities better by avoiding overly aggressive pruning.

To determine an upper bound of correctness for the prototype, we inspected the PG representations to see how many alternative sets contained the correct interpretation even if not the highest scoring alternative. This number is different from the correctness score in Fig. 6, which is the percentage of gold standard triples that are the highest scoring alternatives in the merged PG.

	baseline	prototype
nodes w/ the correct type	76	**91**
edges w/ the correct relation	74	**88**

Table 1: Percentage of nodes and edges containing the correct types and semantic relations in the baseline and the prototype for all 37 sentences.

Table. 1 shows that 91% of the nodes in the PG contain the correct type (though not necessarily the highest scoring). 88% of the edges contain the correct semantic relations among the alternatives. In contrast, the baseline has pruned away 24% of the correct types and 26% of the correct semantic relations.

7 Experiment 2

Our second experiment aims to evaluate the claim that the prototype can efficiently manage a large number of alternative interpretations. The top line in Fig. 7 shows the number of triples in the PG representations input to the prototype. This is the total number of triples (including ambiguous alternatives) in the PG for each sentence prior to invoking Algorithm 1. The middle line is the number of triples remaining after merging and pruning by Algorithm 1. The bottom line is the number of triples after pruning all but the highest scoring alternatives (the baseline system). The results show that Algorithm 1 achieves significant compression over unmerged PG representations. The resulting size of the merged PG representations more closely tracks the size of the aggressively pruned

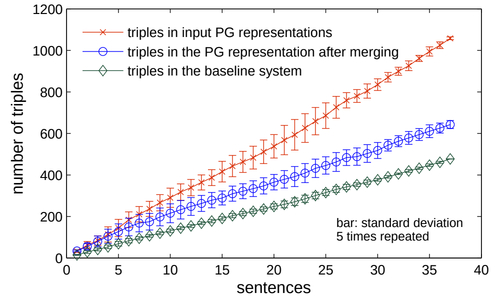


Figure 7: Total number of triples in individual sentence PG representations (top); total number of triples in the PG representation after merging in the prototype system (middle); total number of triples after pruning to the highest scoring alternative (bottom).

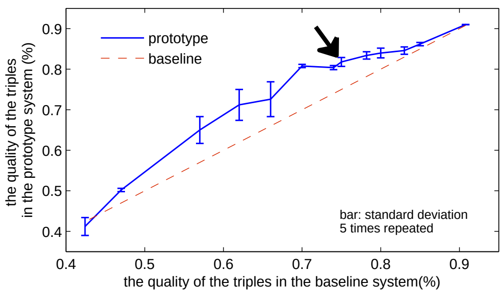


Figure 8: Sensitivity of the prototype and baseline systems to the quality of the NL system output. The quality of input triples is perturbed affecting performance accuracy of the two systems. For example, when the quality of input triples is such that the baseline system performs at 70% accuracy, the prototype system performs at 80%. The arrow indicates unperturbed language interpreter performance.

representations.

8 Experiment 3

Finally, we wanted to measure the sensitivity of our approach to the quality of the natural language interpretation. In this experiment, we artificially varied the confidence scores for the correct interpretations in the PG representations input to the prototype and baseline systems by a fixed percentage. For example, consider a node heart-1 with multiple candidate types, including the correct sense for its context: INTERNAL-ORGAN with confidence 0.8. We reran Experiment 1 varying the confidence in INTERNAL-ORGAN in increments of +/-10%, while scaling the confidences in the incorrect types equally. As the confidence in correct interpretations is increased, all correct interpretations become the highest scoring, so aggressive pruning is justified and the baseline performance approaches the prototype performance. As the confidences in correct interpretations are decreased, they are more likely to be pruned by both systems.

Fig. 8 shows that Algorithm 1 is able to recover at least some correct interpretations even when their original scores (relative to incorrect alternatives) is quite low.

9 Discussion and Future Work

Our controlled experiments suggest that it is both desirable and feasible to delay ambiguity resolution beyond sentence and text boundaries. Improvements in the correctness of semantic interpretation of sentences is possible without an explosion in size when maintaining multiple interpretations.

Nevertheless, these experiments are proofs of concept. The results confirm that it is worthwhile to subject our prototype to a more real-world, practical application. To do so, we need to address several issues.

First, we manually simulated structural (parse) ambiguities. We will complete modifications to the Stanford Parser to produce PG representations natively. This change will result in a significant increase in the number of alternatives stored in the PG representation over the current prototype. Our initial investigations suggest that there is still enough structural overlap among the candidate parse trees to allow the PG representation to control explosion, but this is an empirical question that will need to be confirmed.

We are modifying our semantic interpreter to admit induced semantic interpretation rules which will allow us to train the system in new domains.

The current prototype uses a naive heuristic for identifying co-reference candidates. We are investigating the use of off-the-shelf co-reference systems.

Finally, we are incorporating the Alchemy (Richardson and Domingos, 2006) probabilistic inference engine to calculate the probability that a candidate interpretation is correct given the PG constraints and alignments, in order to inform confirmation or pruning of interpretations.

Once these updates are complete, we will perform more wide-scale evaluations. We will investigate the automatic construction of a test corpus using text clustering to find redundant texts, and we will conduct experiments in multiple domains.

10 Related Work

Succinctly representing multiple interpretations has been explored by several researchers. The packed representation (Maxwell III and Kaplan, 1981; Crouch, 2005) uses logical formulae to denote alternative interpretations and treats the disambiguation task as the propositional satisfiability problem. Core Language Engine (Alshawi, 1992) introduces two types of packing mechanism. First, a quasi logical form allows the underspecification of several types of information, such as anaphoric references, ellipsis and semantic relations (Alshawi and Crouch, 1992). Second, a packed quasi logical form (Alshawi, 1992) compactly represents the derivations of alternative quasi logical forms. In contrast, the PG representation is (1) based on a graphical representation, (2) explicitly represents constraints and (3) includes confidence scores.

These representations and the PG representation have one feature in common: they represent a set of complete alternative interpretations of a text. Another class of compact representations, called "underspecification", has been studied as a formal representation of ambiguous sentences. These representations include Hole Semantics (Bos, 2004), Underspecified Discourse Representation Semantics (Reyle, 1995), Minimal Recursion Semantics (Copestake et al., 2005) and Dominance Constraints (Egg et al., 2001). These representations, rather than packing fully-represented candidate interpretations, specify fragments of interpretations which are unambiguously interpreted, along with constraints on their combination (corresponding to different interpretations). They generally focus on specific ambiguities such as scope ambiguity (Bos, 2004) (Egg et al., 2001) (Copestake et al., 2005) or discourse relations (Schilder, 1998) (Regneri et al., 2008).

Disambiguating compact representations has received relatively less attention. (Riezler et al., 2002; Geman and Johnson, 2002) use a packed representation to train parsers on a corpus and uses the learned statistics to disambiguate packed representations. (Clark and Harrison, 2010) uses paraphrase databases and a hand-built knowledge base to resolve underspecified representations.

Different architectures have been proposed to improve the pipeline architecture. (Sutton and McCallum, 2005; Wellner et al., 2004) maintain a beam of n best interpretations in the pipeline architecture. Their pipeline, however, consists of only two components. (Finkel et al., 2006) uses sampling over the distribution of alternative interpretations at each stage of the pipeline and then passes the sampled data to the next component. The packed representation (Crouch, 2005) and CLE (Alshawi, 1992) use packed representation in the pipeline, though both, at some stages, unpack them and re-pack the processed result. (Crouch and King, 2006) later proposes a new method that does not require unpacking and then repacking.

11 Conclusion

We have begun to address the challenge of efficiently managing multiple alternative interpretations of text. We have presented (1) a *packed graphical representation* that succinctly represents multiple alternative interpretations as well as the constraints among them, and (2) an algorithm for combining multiple PG representations to reinforce correct interpretations and discount implausible interpretations. Controlled experiments show that it is possible to improve the correctness of semantic interpretations of text by delaying disambiguation, without incurring the cost of an exponentially expanding representation.

12 Acknowledgement

Support for this research was provided in part by Air Force Contract FA8750-09-C-0172 under the DARPA Machine Reading Program

References

Alshawi, Hiyan and Richard S. Crouch. 1992. Monotonic semantic interpretation. In *ACL*, pages 32–39.

Alshawi, Hiyan, editor. 1992. *The Core Language Engine*. MIT Press, Cambridge, Massachusetts.

Barker, Ken, Bruce Porter, and Peter Clark. 2001. A library of generic concepts for composing knowledge bases. In *Proceedings of the international conference on Knowledge capture*, pages 14–21.

Bos, Johan. 2004. Computational semantics in discourse: Underspecification, resolution, and inference. *Journal of Logic, Language, and Information*, 13(2):139–157.

Clark, Peter and Phil Harrison. 2010. Exploiting paraphrases and deferred sense commitment to interpret questions more reliably. In *To appear in Proceedings of CoLing 2010*.

Copestake, Ann, Dan Flickinger, Carl Pollard, and Ivan Sag. 2005. Minimal recursion semantics: an introduction. *Research on Language and Computation*, 3:281–332.

Crouch, Richard S. and Tracy Holloway King. 2006. Semantics via f-struecture rewriting. In *Proceedings of LFG06 Conference*.

Crouch, Dick. 2005. Packed rewriting for mapping semantics to kr. In *In Proceedings Sixth International Workshop on Computational Semantics*.

Egg, Markus, Alexander Koller, and Joachim Niehren. 2001. The constraint language for lambda structures. *Journal of Logic, Language, and Information Vol 10 (4), 2001, pp.457-485*, 10:457–485.

Finkel, Jenny Rose, Christopher D. Manning, and Andrew Y. Ng. 2006. Solving the problem of cascading errors: approximate bayesian inference for linguistic annotation pipelines. In *EMNLP*, pages 618–626, Morristown, NJ, USA.

Geman, Stuart and Mark Johnson. 2002. Dynamic programming for parsing and estimation of stochastic unification-based grammars. In *ACL*, pages 279–286.

Klein, Dan and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*, pages 423–430, Morristown, NJ, USA.

Lesk, Michael. 1986. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26, New York, NY, USA.

Maxwell III, John T. and Ronald M. Kaplan. 1981. A method for disjunctive constraint satisfaction. In Tomita, Masaru, editor, *Current Issues in Parsing Technology*, pages 173–190. Kluwer Academic Publishers, Dordrecht.

Patwardhan, Siddharth, Satanjeev Banerjee, and Ted Pedersen. 2005. Senserelate:: Targetword-A generalized framework for word sense disambiguation. In *ACL*.

Pearl, Judea. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan-Kaufmann.

Punyakanok, Vasin, Dan Roth, and Wen tau Yih. 2005. The necessity of syntactic parsing for semantic role labeling. In Kaelbling, Leslie Pack and Alessandro Saffiotti, editors, *IJCAI*, pages 1117–1123. Professional Book Center.

Regneri, Michaela, Markus Egg, and Alexander Koller. 2008. Efficient processing of underspecified discourse representations. In *HLT*, pages 245–248, Morristown, NJ, USA.

Reyle, Uwe. 1995. Underspecified discourse representation structures and their logic. *Logic Journal of the IGPL*, 3(2-3):473–488.

Richardson, Matthew and Pedro Domingos. 2006. *Markov logic networks*. Kluwer Academic Publishers.

Riezler, Stefan, Tracy H. King, Ronald M. Kaplan, Richard S. Crouch, John T. Maxwell III, and Mark Johnson. 2002. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *ACL*, pages 271–278.

Schilder, Frank. 1998. An underspecified segmented discourse representation theory (USDRT). In *COLING-ACL*, pages 1188–1192.

Sutton, Charles and Andrew McCallum. 2005. Joint parsing and semantic role labeling. In *CONLL*, pages 225–228, Morristown, NJ, USA.

Wellner, Ben, Andrew McCallum, Fuchun Peng, and Michael Hay. 2004. An integrated, conditional model of information extraction and coreference with application to citation matching. In *UAI*, pages 593–601, Arlington, Virginia, United States.

Automatic generation of inter-passage links based on semantic similarity

Petr Knoth, Jakub Novotny, Zdenek Zdrahal
Knowledge Media Institute
The Open University
p.knoth@open.ac.uk

Abstract

This paper investigates the use and the prediction potential of semantic similarity measures for automatic generation of links across different documents and passages. First, the correlation between the way people link content and the results produced by standard semantic similarity measures is investigated. The relation between semantic similarity and the length of the documents is then also analysed. Based on these findings a new method for link generation is formulated and tested.

1 Introduction

Text retrieval methods are typically designed to find documents relevant to a query based on some criterion, such as BM25 or cosine similarity (Manning et al., 2008). Similar criteria have also been used to identify documents relevant to the given reference document, thus in principle linking the reference document to the related documents (Wilkinson and Smeaton, 1999). This paper studies the correspondence between the results of this approach and the way linking is performed by people. The study confirms that the length of documents is an important factor usually causing the quality of current link generation approaches to deteriorate. As a result, methods working at a finer granularity than documents should be investigated. This will also improve the speed of access to information. For example, when users read through a long document, they should be able to quickly access a passage in another possibly long document related to the discussed topic. The automatic detection of document pairs containing highly related passages is the task addressed in this paper.

This work has been partially supported by Eurogene - Contract no. ECP-2006-EDU-410018)

A number of approaches for automatic link generation have used measures of semantic similarity. While these measures were widely used for the discovery of related documents in practise, their correspondence to the way people link content has not been sufficiently investigated (see Section 2). As our contribution to this topic, we present in this paper an approach which tries to first investigate this correspondence on a large text corpus. The resulting method is then motivated by the outcomes of this analysis.

It has been recognised in information retrieval that when a collection contains long documents, better performance is often achieved by breaking each document into subparts or passages and comparing these rather than the whole documents to a query (Manning et al., 2008). A suitable granularity of the breakdown is dependent on a number of circumstances, such as the type of the document collection or the information need. In this work, we have decided to work at the level of documents and paragraphs. Our task can be formalized as a two-step process:

1. Given a collection of documents, our goal is to identify candidate pairs of documents between which a link may be induced.

2. Given each candidate pair of documents, our task is to identify pairs of passages, such that the topics in the passages are related in both documents.

The method presented in this paper has many

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 590–598,
Beijing, August 2010

potential applications. First, it may be used for the interlinking of resources that were not originally created as hypertext documents and for the maintenance or the discovery of new links as the collection grows. Second, the method can be applied to improve navigation in collections with long texts, such as books or newspaper articles. A link may be identified by the system automatically and the user can be pointed immediately to the part of the text which is relevant to the block of text currently being read. Similar application has been developed by (Kolak and Schilit, 2008) who provided a method for mining repeated word sequences (quotations) from very large text collections and integrated it with the Google Books archive. Other application areas may involve text summarization and information retrieval.

The paper makes the following contributions:

- It provides a new interpretation and insight in the use of semantic similarity measures for the automatic generation of links.
- It develops a novel two-step approach for the discovery of passage-passage links across potentially long documents and it identifies and discusses the selection of the parameters.

The rest of the paper is organized as follows. Section 2 presents the related work in the field. Section 3 discusses the data selected for our experiment and Section 4 describes how the data were processed in order to perform our investigation. In Section 5, the analysis in which we compared the results produced by semantic similarity measures with respect to the way people link content is presented. Section 6 then draws on this analysis and introduces the method for automatic generation of links which is finally evaluated in Section 7.

2 Related Work

In the 1990s, the main application area for link generation methods were hypertext construction systems. A survey of these methods is provided by (Wilkinson and Smeaton, 1999). In the last decade, methods for finding related documents became the de-facto standard in large digital repositories, such as PubMed or the ACM Digital Library. Search engines including Google also generate links to related pages or research articles.

Generating links pointing to units of a smaller granularity than a document, which can be considered as a task of *passage* or *focused* retrieval, has also been addressed recently. In this task, the system locates the relevant information inside the document instead of only providing a link to the document. The Initiative for the Evaluation of XML retrieval (INEX) started to play an essential role in link generation by providing tracks for the evaluation of link generation systems (Huang et al., 2008; Huang et al., 2009) using the Wikipedia collection at both the document and the passage level.

Current approaches can be divided into three groups: (1) *link-based* approaches discover new links by exploiting an existing link graph (Itakura and Clarke, 2008; Jenkinson et al., 2008; Lu et al., 2008). (2) *semi-structured* approaches try to discover new links using semi-structured information, such as the anchor texts or document titles (Geva, 2007; Dopichaj et al., 2008; Granitzer et al., 2008). (3) *purely content-based* approaches use as an input plain text only. They typically discover related resources by calculating semantic similarity based on document vectors (Allan, 1997; Green, 1998; Zeng and Bloniarz, 2004; Zhang and Kamps, 2008; He, 2008). Some of the mentioned approaches, such as (Lu et al., 2008), combine multiple approaches.

Although link generation methods are widely used in practise, more work is needed to understand which features contribute to the quality of the generated links. Work in this area includes the study of (Green, 1999) who investigated how lexical chaining based on ontologies can contribute to the quality of the generated links, or the experiments of (Zeng and Bloniarz, 2004) who compared the impact of the manually and automatically extracted keywords. There has also been effort in developing methods that can in addition to link generation assign a certain semantic type to the extracted links and thus describe the relationship between documents (Allan, 1997).

The method presented in this paper is purely content-based and therefore is applicable in any text collection. Its use in combination with link-based or semi-structured approaches is also possible. The rationale for the method comes from the analysis of the prediction potential of semantic similarity for automatic link generation presented in Section 5. Related analysis is presented in (He, 2008) which claims that linked articles are more likely to be semantically similar[1], however, the study does not provide sufficient evidence to confirm and describe this relationship. In link generation, we are more interested in asking the opposite question, i.e. whether articles with higher semantic similarity are more likely to be linked. Our study provides a new insight into this relationship and indicates that the relationship is in fact more complex than originally foreseen by He.

3 Data selection

This section introduces the document collection used for the analysis and the experiments. The following properties were required for the document collection to be selected for the experiments. First, in order to be able to measure the correlation between the way people link content and the results produced by semantic similarity measures, it was necessary to select a document collection which can be considered as relatively well interlinked. Second, it was important for us to work with a collection containing a diverse set of topics. Third, we required the collection to contain articles of varied length. We were mostly interested in long documents, which create conditions for the testing of passage retrieval methods. We decided to use the Wikipedia collection, because it satisfies all our requirements and has also been used in the INEX Link-The-Wiki-Track.

Wikipedia consists of more than four million pages spread across five hundred thousands categories. As it would be for our calculation unnecessarily expensive to work with the whole encyclopedia, a smaller, but still a sufficiently large subset of Wikipedia, which satisfies our requirements of topic diversity and document length, was selected. Our document collection was generated from articles in categories containing the words United Kingdom. This includes categories, such as United Kingdom, Geography of United Kingdom or History of the United Kingdom. There are about 3,000 such categories and 57,000 distinct articles associated to them. As longer articles provide better test conditions for passage retrieval methods, we selected the 5,000 longest articles out of these 57,000. This corresponds to a set where each article has the length of at least 1,280 words.

[1]With respect to the cosine similarity measure.

4 Data preprocessing

Before discussing the analysis performed on the document collection, let us briefly describe how the documents were processed and the semantic similarity calculated.

First, the N articles/documents $D = \{d_1, d_2, \dots, d_N\}$ in our collection were preprocessed to extract plain text by removing the Wiki markup. The documents were then tokenized and a dictionary of terms $T = \{t_1, t_2, \dots, t_M\}$ was created. Assuming that the order of words can be neglected (the bag-of-words assumption) the document collection can be represented using a $N \times M$ term-document matrix. In this way, each document is modelled as a vector corresponding to a particular row of the matrix. As it is inefficient to represent such a sparse vector in memory (most of the values are zeros), only the non-zero values were stored. *Term frequency - inverse document frequency (tfidf)* weighting was used to calculate the values of the matrix. Term frequency tf_{t_i,d_j} is a normalized frequency of term t_i in document d_j:

$$tf_{t_i,d_j} = \frac{f(t_i, d_j)}{\sum_k f(t_k, d_j)}$$

Inverse document frequency idf_{t_i} measures the general importance of term t_i in the collection of documents D by counting the number of documents which contain term t_i:

$$idf_{t_i} = \log \frac{|D|}{|d_j : t_i \in d_j|}$$

$$tfidf_{t_i,d_j} = tf_{t_i,d_j}.idf_{t_i}$$

Similarity is then defined as the function $sim(\overrightarrow{x}, \overrightarrow{y})$ of the document vectors $\overrightarrow{x}$ and $\overrightarrow{y}$. There exists a number of similarity measures used for the calculation of similarity between two vectors (Manning and Schuetze, 1999), such as *cosine*, *overlap*, *dice* or *Jaccard* measures. Some studies employ algorithms for the reduction of dimensions of the vectors prior to the calculation of similarity to improve the results. These approaches may involve techniques, such as lexical chaining (Green, 1999), Latent Semantic Indexing (Deerwester et al., 1990), random indexing (Widdows and Ferraro, 2008) and Latent Dirichlet Allocation (Blei et al., 2003). In this work we intentionally adopted perhaps the most standard similarity measure - cosine similarity calculated on the *tfidf* vectors and no dimensionality reduction technique was used. The formula is provided for completeness:

$$sim_{cosine}(\overrightarrow{x}, \overrightarrow{y}) = \frac{\overrightarrow{x}.\overrightarrow{y}}{|x|.|y|}$$

Cosine similarity with *tfidf* vectors has been previously used in automatic link generation systems producing state-of-the-art results when compared to other similarity measures (Chen et al., 2004). This allows us to report on the effectiveness of the most widely used measure with respect to the way the task is completed by people. While more advanced techniques might be in some cases better predictors for link generation, we did not experiment with them as we preferred to focus on the investigation of the correlation between the most widely used measure and manually created links. Such study has to our knowledge never been done before, but it is necessary for the justification of automatic link generation methods.

5 Semantic similarity as a predictor for link generation

The document collection described in Section 3 has been analysed as follows. First, pair-wise similarities using the formulas described in Section 4 were calculated. Cosine similarity is a symmetric function and, therefore, the calculation of all inter-document similarities in the dataset of $5,000$ documents requires the evaluation of $\frac{5,000^2}{2} - 5,000 = 12,495,000$ combinations. Figure 1 shows the distribution of the document pairs (on a log_{10} scale) with respect to their similarity value. The frequency follows a power law distribution. In our case, 99% of the pairs have similarity lower than 0.1.

To compare the semantic similarity measures with the links created by Wikipedia authors, all inter-document intra-collection links, i.e. links

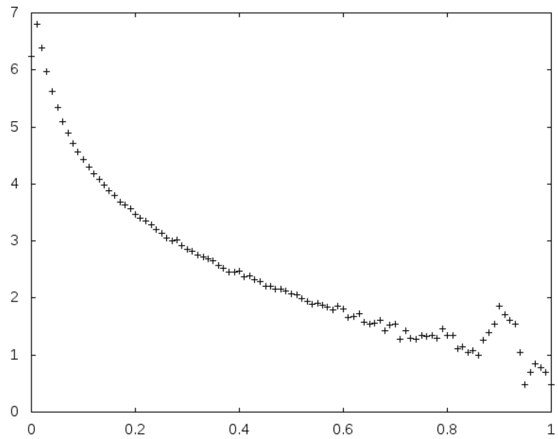


Figure 1: The histogram shows the number of document pairs on a $\log_{10}$ scale (y-axis) with respect to their cosine similarity (x-axis).

created by users of Wikipedia commencing from and pointing to a document within our collection, were extracted. These links represent the connections as seen by the users regardless of their direction. Each of these links can be associated with a similarity value calculated in the previous step. Documents with similarity lower than 0.1 were ignored. Out of the $120,602$ document pairs with inter-document similarity higher than 0.1, $17,657$ pairs were also connected by a user-created link.

For the evaluation, interval with cosine similarity $[0.1, 1]$ was divided evenly into 100 buckets and all 120,602 document pairs were assigned to the buckets according their similarity values. From the distribution shown in Figure 1, buckets corresponding to higher similarity values contain fewer document pairs than buckets corresponding to smaller similarity values. Therefore, for each bucket, the number of user created links within the bucket was normalized by the number of document pairs in the bucket. This number is the likelihood of the document pair being linked and will be called *linked-pair likelihood*. The relation between semantic similarity and linked-pair likelihood is shown in Figure 2.

As reported in Section 2, semantic similarity has been previously used as a predictor for the automatic generation of links. The typical scenario was that the similarity between pairs of documents was calculated and the links between the

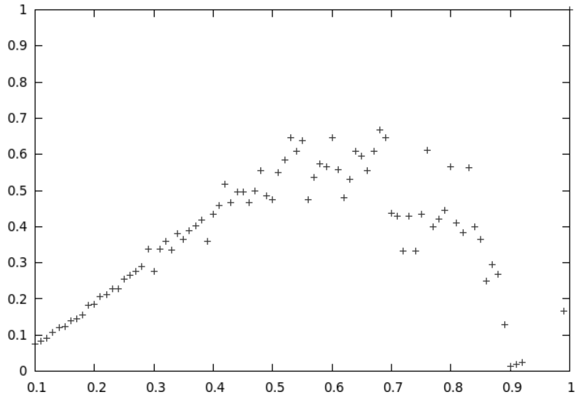


Figure 2: The linked-pair likelihood (y-axis) with respect to the cosine similarity (x-axis).

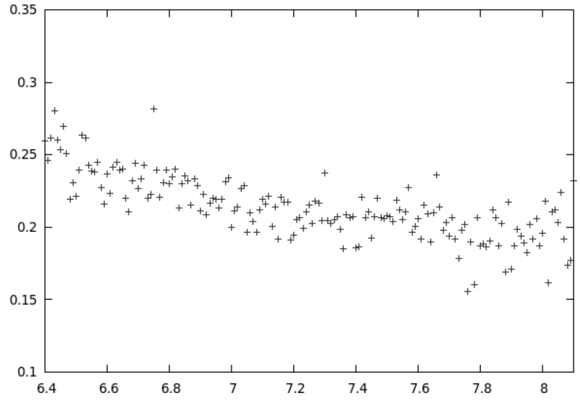


Figure 3: The average cosine similarity (y-axis) of document pairs of various length (x-axis) between which there exists a link. The x-axis is calculated as a $\log_{10}(l_1.l_2)$

most similar documents were generated (Wilkinson and Smeaton, 1999). If this approach was correct, we would expect the curve shown in Figure 2 to be monotonically increasing. However, the relation shown in Figure 2 is in accordance with our expectations only up to the point 0.55. For higher values of inter-document similarity the linked-pair likelihood does not rise or it even decreases.

Spearman's rank correlation and Pearson correlation were applied to estimate the correlation coefficients and to test the statistical significance of our observation. This was performed in two intervals: $[0, 0.55]$ and $[0.55, 1]$. A very strong positive correlation 0.986 and 0.987 have been received in the first interval for the Spearman's and Pearson coefficients respectively. A negative correlation -0.640 and -0.509 have been acquired for the second interval again for the Spearman's and Pearson coefficients respectively. All the measured correlations are significant for p-value well beyond $p < 0.001$. Very similar results have been achieved using different collections of documents.

The results indicate that high similarity value is not necessarily a good predictor for automatic link generation. A possible explanation for this phenomenon is that people create links between related documents that provide new information and therefore do not link nearly identical content. However, as content can be in general linked for various purposes, more research is needed to investigate if document pairs at different similarity levels also exhibit different qualitative properties. More specifically, can the value of semantic similarity be used as a predictor for relationship typing?

An important property of semantic similarity as a measure for automatic generation of links is the robustness with respect to the length of documents. As mentioned in Section 4, cosine similarity is by definition normalized by the product of the documents length. Ideally the cosine similarity should be independent of the documents length. To verify this in our dataset, we have taken pairs of documents between which Wikipedia users assigned links and divided them into buckets with respect to the function $\log_{10}(l_1.l_2)$, where l_1 and l_2 are the lengths of the two documents in the document pair and the logarithm is used for scaling. The value of each bucket was calculated as an average similarity of the bucket members. The results are shown in Figure 3. The graph shows that the average similarity value is slightly decreasing with respect to the length of the articles. Values -0.484 and -0.231 were obtained for Spearman's and Pearson correlation coefficients respectively. Both correlations are statistically significant for $p < 0.001$. A much stronger correlation was measured for Spearman's than for Pearson which can be explained by the fact that Spearman's correlation is calculated based on ranks rather than real values and is thus less sensitive to outliers.

Our experience from repeating the same experiment on another Wikipedia subset generated from categories containing the word Geography tells us that the decrease is even more noticeable when short and long articles are combined. The decrease in average similarity suggests that if cosine similarity is used for the automatic generation of links then document pairs with higher value of $l_1.l_2$ have a higher linked-pair likelihood than pairs with a smaller value of this quantity. In other words, links created between documents with small $l_1.l_2$ typically exhibit a larger value of semantic similarity than links created between documents with high value of $l_1.l_2$. Although the decrease may seem relatively small, we believe that this knowledge may be used for improving automatic link generation methods by adaptively modifying the thresholds with respect to the $l_1.l_2$ length.

6 Link generation method

In this section we introduce the method for the automatic generation of links. The method can be divided into two parts (1) Identification of candidate link pairs (i.e. the generation of document-to-document links) (2) Recognition of passages sharing a topic between the two documents (i.e. the generation of passage-to-passage links).

6.1 Document-to-document links

The algorithm for link generation at the granularity of a document is motivated by the findings reported in Section 5.

Algorithm 1: Generate document links
Input: A set of document vectors D,
min. sim. α, max. sim. $\beta \in [0, 1], C = \emptyset$
Output: A set C of candidate links
of form $\langle d_i, d_j, sim \rangle \in C$ where d_i and d_j are
documents and $sim \in [0, 1]$ is their similarity
1.**for each** $\{\langle d_i, d_j \rangle | i, j \in \aleph_0 \wedge i < j < |D|\}$ **do**
2. $sim_{d_i,d_j} := similarity(d_i, d_j)$
3. **if** $sim_{d_i,d_j} > \alpha \wedge sim_{d_i,d_j} < \beta$ **then**
4. $C := C \cup \langle d_i, d_j, sim_{d_i,d_j} \rangle$

The algorithm takes as the input a set of document vectors and two constants - the minimum and maximum similarity thresholds - and iterates over all pairs of document vectors. It outputs all document vector pairs, such that their similarity is higher than α and smaller than β. For well chosen β, the algorithm does not generate links between nearly duplicate pairs. If we liked to rank the discovered links according to the confidence of the system, we would suggest to assign each pair a value using the following function.

$$rank_{d_i,d_j} = |sim_{d_i,d_j} - (\alpha + \frac{\beta - \alpha}{2})|$$

The ranking function makes use of the fact that the system is most confident in the middle of the similarity region defined by constants α and β, under the assumption that suitable values for these constants are used. The higher the rank of a document pair, the better the system's confidence.

6.2 Passage-to-passage links

Due to a high number of combinations, it is typically infeasible even for relatively small collections to generate passage-to-passage links across documents directly. However, the complexity of this task is substantially reduced when passage-to-passage links are discovered in a two-step process.

Algorithm 2: Generate passage links
Input: Sets P_i, P_j of paragraph document
vectors for each pair in C
min. sim. γ, max. sim. $\delta \in [0, 1]$ such that
$\alpha < \gamma \wedge \beta < \delta, , L = \emptyset$
Output: A set L of passage links
of form $\langle p_{k_i}, p_{l_j}, sim \rangle \in L$ where p_{k_i} and
p_{l_j} are paragraphs in documents d_i, d_j
and $sim \in [0, 1]$ is their similarity
1.**for each** $\{\langle p_{k_i}, p_{l_j} \rangle | p_{k_i} \in P_i, p_{l_j} \in P_j\}$ **do**
2. $sim_{p_{k_i},p_{l_j}} := similarity(p_{k_i}, p_{l_j})$
3. **if** $sim_{p_{k_i},p_{l_j}} > \gamma \wedge sim_{p_{k_i},p_{l_j}} < \delta$ **then**
4. $L := L \cup \langle p_{k_i}, p_{l_j}, sim_{p_{k_i},p_{l_j}} \rangle$

As Section 5 suggests, the results of Algorithm 1 may be improved by adaptive changing of the thresholds α and β based on the length of the document vectors. More precisely, in the case of cosine similarity, this is the quantity $lr = l_1.l_2$. The value α should be higher (β lower) for pairs with low lr than for pairs with high lr and vice versa. Although the relative quantification of this ratio is left for future work, we believe that we can exploit these findings for the generation of passage-to-passage links.

More specifically, we know that the length of passages (paragraphs in our case) is lower than the length of the whole documents. Hence, the similarity of a linked passage-to-passage pair should be on average higher than the similarity of a linked document-to-document pair, as revealed by the results of our analysis. This knowledge is used within Algorithm 2 to set the parameters γ and δ. The algorithm shows, how passage-to-passage links are calculated for a single document pair previously identified by Algorithm 1. Applying the two-step process allows the discovery of document pairs, which are likely to contain strongly linked passages, at lower similarity levels and to recognize the related passages at higher similarity levels while still avoiding duplicate content.

7 Results

The experimental evaluation of the methods presented in Section 6 is divided into two parts: (1) the evaluation of document-to-document links (Algorithm 1) and (2) the evaluation of passage-to-passage links (Algorithm 2).

7.1 Evaluation of document-to-document links

As identified in Section 5 (and shown in Figure 2), the highest linked-pair likelihood does not occur at high similarity values, but rather somewhere between similarity 0.5 and 0.7. According to Figure 2, the linked-pair likelihood in this similarity region ranges from 60% to 70%. This value is in our view relatively high and we think that it can be explained by the fact that Wikipedia articles are under constant scrutiny by users who eventually discover most of the useful connections. However, how many document pairs that could be linked in this similarity region have been missed by the users? That is, how much can our system help in the discovery of possible connections?

Suppose that our task would be to find document pairs about linking of which the system is most certain. In that case we would set the thresholds α and β somewhere around these values depending on how many links we would like to obtain. In our evaluation, we have extracted pairs of documents from the region between $\alpha = 0.65$ and $\beta = 0.70$ regardless of whether there originally was a link assigned by Wikipedia users. An evaluation tool which allowed a subject to display the pair of Wiki documents next to each other and to decide whether there should or should not be a link between the documents was then developed. We did not inform the subject about the existence or non-existence of links between the pages. More specifically, the subject was asked to decide yes (link generated correctly) if and only if they found it beneficial for a reader of the first or the second article to link them together regardless of the link direction. The subject was asked to decide no (link generated incorrectly) if and only if they felt that navigating the user from or to the other document does not provide additional value. For example, in cases where the relatedness of the documents is based on their lexical rather than their semantic similarity.

The study revealed that 91% of the generated links were judged by the subject as correct and 9% as incorrect. Table 1 shows the results of the experiment with respect to the links originally assigned by the users of Wikipedia. It is interesting to notice that in 3% of the cases the subject decided not to link the articles even though they were in fact linked on Wikipedia. Overall, the algorithm discovered in 30% of the cases a useful connection which was missing in Wikipedia. This is in line with the findings of (Huang et al., 2008) who claims that the validity of existing links in Wikipedia is sometimes questionable and useful links may be missing.

An interesting situation in the evaluation occurred when the subject discovered a pair of articles with titles *Battle of Jutland* and *Night Action at the Battle of Jutland*. The Wikipedia page indicated that it is an orphan and asked users of Wikipedia to link it to other Wikipedia articles. Our method would suggest the first article as a good choice.

		Wikipedia link	
		yes	no
Subject's decision	yes	0.61	0.30
	no	0.03	0.06

Table 1: Document-to-document links from the $[0.65, 0.7]$ similarity region. The subject's decision in comparison to the Wikipedia links.

		Wikipedia link	
		yes	no
Subject's decision at page level	yes	0.16	0.10
	no	0.18	0.56

Table 2: Document-to-document candidate links generation from the $[0.2, 0.21]$ similarity region and document pairs with high lr ($lr \in [7.8 - 8]$).

7.2 Evaluation of passage-to-passage linking

The previous section provided evidence that the document-to-document linking algorithm is capable of achieving high performance when parameters α, β are well selected. However, Section 5 indicated that it is more difficult to discover links across long document pairs. Thereby, we have evaluated the passage-to-passage linking on document pairs with quite low value of similarity $[0.2, 0.21]$. According to Figure 2, this region has only 15% linked-pair likelihood.

Clearly, our goal was not to evaluate the approach in the best possible environment, but rather to check whether the method is able to discover valuable passage-to-passage links from very long articles with low similarity. Articles with this value of similarity would be typically ranked very poorly by link generation methods working at the document level.

Table 2 shows the results after the first step of the approach, described in Section 6, with respect to the links assigned by Wikipedia users. As in the previous experiment, the subject was given pairs of documents and decided whether they should or should not be linked. Parameters α and β were set to 0.2, 0.21 respectively. Table 2 indicates that that the accuracy $(16\% + 10\% = 26\%)$ is at this similarity region much lower than the one reported in Table 1, which is exactly in line with our expectations. It should be noticed that 34% of the document pairs were linked by Wikipedia users, even though only 15% would be predicted by linked-pair likelihood shown in Figure 2. This confirms that long document pairs exhibit a higher probability of being linked in the same similarity region than shorter document pairs.

If our approach for passage-to-passage link generation (Algorithm 2) is correct, we should be able to process the documents paragraphs and detect possible passage-to-passage links. The selection of the parameters γ and δ influences the willingness of the system to generate links. For this experiment, we set the parameters γ, δ to 0.4, 0.8 respectively. The subject was asked to decide: (1) if the connection discovered by the link generation method at the granularity of passages was useful (when the system generated a link) (2) whether the decision not to generate link is correct (when the system did not generate a link). The results of this evaluation are reported in Table 3. It can be seen that the system made in 60% $(14\% + 46\%)$ of the cases the correct decision. Most mistakes were made by generating links that were not sufficiently related (24%). This might be improved by using a higher value of γ (lower value of δ).

		System's decision	
		yes	no
Subject's decision	yes (correct)	0.14	0.46
	no (incorrect)	0.24	0.16

Table 3: Passage-to-passage links generation for very long documents. Passages extracted from the $[0.4, 0.8]$ similarity region.

8 Conclusions

This paper provided a new insight into the use of semantic similarity as a predictor for automatic link generation by performing an investigation in the way people link content. This motivated us in the development of a novel purely content-based approach for automatic generation of links at the granularity of both documents and paragraphs which does not expect semantic similarity and linked-pair likelihood to be directly proportional.

References

Allan, James. 1997. Building hypertext using information retrieval. *Inf. Process. Manage.*, 33:145–159, March.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *JOURNAL OF MACHINE LEARNING RESEARCH*, 3:993–1022.

Chen, Francine, Ayman Farahat, and Thorsten Brants. 2004. Multiple similarity measures and source-pair information in story link detection. In *In HLT-NAACL 2004*, pages 2–7.

Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407.

Dopichaj, Philipp, Andre Skusa, and Andreas Heß. 2008. Stealing anchors to link the wiki. In Geva et al. (Geva et al., 2009), pages 343–353.

Geva, Shlomo, Jaap Kamps, and Andrew Trotman, editors. 2009. *Advances in Focused Retrieval, 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008, Dagstuhl Castle, Germany, December 15-18, 2008. Revised and Selected Papers*, volume 5631 of *Lecture Notes in Computer Science*. Springer.

Geva, Shlomo. 2007. Gpx: Ad-hoc queries and automated link discovery in the wikipedia. In Fuhr, Norbert, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors, *INEX*, volume 4862 of *Lecture Notes in Computer Science*, pages 404–416. Springer.

Granitzer, Michael, Christin Seifert, and Mario Zechner. 2008. Context based wikipedia linking. In Geva et al. (Geva et al., 2009), pages 354–365.

Green, Stephen J. 1998. Automated link generation: can we do better than term repetition? *Comput. Netw. ISDN Syst.*, 30(1-7):75–84.

Green, Stephen J. 1999. Building hypertext links by computing semantic similarity. *IEEE Trans. on Knowl. and Data Eng.*, 11(5):713–730.

He, Jiyin. 2008. Link detection with wikipedia. In Geva et al. (Geva et al., 2009), pages 366–373.

Huang, Wei Che, Andrew Trotman, and Shlomo Geva. 2008. Experiments and evaluation of link discovery in the wikipedia.

Huang, Wei Che, Shlomo Geva, and Andrew Trotman. 2009. Overview of the inex 2009 link the wiki track.

Itakura, Kelly Y. and Charles L. A. Clarke. 2008. University of waterloo at inex 2008: Adhoc, book, and link-the-wiki tracks. In Geva et al. (Geva et al., 2009), pages 132–139.

Jenkinson, Dylan, Kai-Cheung Leung, and Andrew Trotman. 2008. Wikisearching and wikilinking. In Geva et al. (Geva et al., 2009), pages 374–388.

Kolak, Okan and Bill N. Schilit. 2008. Generating links by mining quotations. In *HT '08: Proceedings of the nineteenth ACM conference on Hypertext and hypermedia*, pages 117–126, New York, NY, USA. ACM.

Lu, Wei, Dan Liu, and Zhenzhen Fu. 2008. Csir at inex 2008 link-the-wiki track. In Geva et al. (Geva et al., 2009), pages 389–394.

Manning, Christopher D. and Hinrich Schuetze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, 1 edition, June.

Manning, Ch. D., P. Raghavan, and H. Schütze. 2008. *Introduction to Information Retrieval*. Cambridge, July.

Widdows, Dominic and Kathleen Ferraro. 2008. Semantic vectors: a scalable open source package and online technology management application. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Bente Maegaard Joseph Mariani Jan Odjik Stelios Piperidis Daniel Tapias, editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

Wilkinson, Ross and Alan F. Smeaton. 1999. Automatic link generation. *ACM Computing Surveys*, 31.

Zeng, Jihong and Peter A. Bloniarz. 2004. From keywords to links: an automatic approach. *Information Technology: Coding and Computing, International Conference on*, 1:283.

Zhang, Junte and Jaap Kamps. 2008. A content-based link detection approach using the vector space model. In Geva et al. (Geva et al., 2009), pages 395–400.

Dependency-driven Anaphoricity Determination for Coreference Resolution

Fang Kong Guodong Zhou Longhua Qian Qiaoming Zhu[*]
JiangSu Provincial Key Lab for Computer Information Processing Technology
School of Computer Science and Technology Soochow University
{kongfang, gdzhou, qianlonghua, qmzhu}@suda.edu.cn

Abstract

This paper proposes a dependency-driven scheme to dynamically determine the syntactic parse tree structure for tree kernel-based anaphoricity determination in coreference resolution. Given a full syntactic parse tree, it keeps the nodes and the paths related with current mention based on constituent dependencies from both syntactic and semantic perspectives, while removing the noisy information, eventually leading to a dependency-driven dynamic syntactic parse tree (D-DSPT). Evaluation on the ACE 2003 corpus shows that the D-DSPT outperforms all previous parse tree structures on anaphoricity determination, and that applying our anaphoricity determination module in coreference resolution achieves the so far best performance.

1 Introduction

Coreference resolution aims to identify which noun phrases (NPs, or mentions) refer to the same real-world entity in a text. According to Webber (1979), coreference resolution can be decomposed into two complementary sub-tasks: (1) anaphoricity determination, determining whether a given NP is anaphoric or not; and (2) anaphor resolution, linking together multiple mentions of a given entity in the world. Although machine learning approaches have performed reasonably well in coreference resolution without explicit anaphoricity determination (e.g. Soon et al. 2001; Ng and Cardie 2002b; Yang et al. 2003, 2008; Kong et al. 2009), knowledge of NP anaphoricity is expected to much improve the performance of a coreference resolution system, since a non-anaphoric NP does not have an antecedent and therefore does not need to be resolved.

Recently, anaphoricity determination has been drawing more and more attention. One common approach involves the design of some heuristic rules to identify specific types of non-anaphoric NPs, such as pleonastic *it* (e.g. Paice and Husk 1987; Lappin and Leass 1994, Kennedy and Boguraev 1996; Denber 1998) and definite descriptions (e.g. Vieira and Poesio 2000). Alternatively, some studies focus on using statistics to tackle this problem (e.g., Bean and Riloff 1999; Bergsma et al. 2008) and others apply machine learning approaches (e.g. Evans 2001;Ng and Cardie 2002a, 2004,2009; Yang et al. 2005; Denis and Balbridge 2007; Luo 2007; Finkel and Manning 2008; Zhou and Kong 2009).

As a representative, Zhou and Kong (2009) directly employ a tree kernel-based method to automatically mine the non-anaphoric information embedded in the syntactic parse tree. One main advantage of the kernel-based methods is that they are very effective at reducing the burden of feature engineering for structured objects. Indeed, the kernel-based methods have been successfully applied to mine structured information in various NLP applications like syntactic parsing (Collins and Duffy, 2001; Moschitti, 2004), semantic relation extraction (Zelenko et al., 2003; Zhao and Grishman, 2005; Zhou et al. 2007; Qian et al., 2008), semantic role labeling (Moschitti, 2004); coreference resolution (Yang et al., 2006; Zhou et al., 2008). One of the key problems for the kernel-based methods is how to effectively capture the structured information according to the nature of the structured object in the specific task.

This paper advances the state-of-the-art performance in anaphoricity determination by ef-

[*] Corresponding author

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 599–607,
Beijing, August 2010

fectively capturing the structured syntactic information via a tree kernel-based method. In particular, a dependency-driven scheme is proposed to dynamically determine the syntactic parse tree structure for tree kernel-based anaphoricity determination by exploiting constituent dependencies from both the syntactic and semantic perspectives to keep the necessary information in the parse tree as well as remove the noisy information. Our motivation is to employ critical dependency information in constructing a concise and effective syntactic parse tree structure, specifically targeted for tree kernel-based anaphoricity determination.

The rest of this paper is organized as follows. Section 2 briefly describes the related work on both anaphoricity determination and exploring syntactic parse tree structures in related tasks. Section 3 presents our dependency-driven scheme to determine the syntactic parse tree structure. Section 4 reports the experimental results. Finally, we conclude our work in Section 5.

2 Related Work

This section briefly overviews the related work on both anaphoricity determination and exploring syntactic parse tree structures.

2.1 Anaphoricity Determination

Previous work on anaphoricity determination can be broadly divided into three categories: heuristic rule-based (e.g. Paice and Husk 1987;Lappin and Leass 1994; Kennedy and Boguraev 1996; Denber 1998; Vieira and Poesio 2000; Cherry and Bergsma 2005), statistics-based (e.g. Bean and Riloff 1999; Cherry and Bergsma 2005; Bergsma et al. 2008) and learning-based methods (e.g. Evans 2001; Ng and Cardie 2002a; Ng 2004; Yang et al. 2005; Denis and Balbridge 2007; Luo 2007; Finkel and Manning 2008; Zhou and Kong 2009; Ng 2009).

The heuristic rule-based methods focus on designing some heuristic rules to identify specific types of non-anaphoric NPs. Representative work includes: Paice and Husk (1987), Lappin and Leass (1994) and Kennedy and Boguraev (1996). For example, Kennedy and Boguraev (1996) looked for modal adjectives (e.g. "necessary") or cognitive verbs (e.g. "It is thought that…" in a set of patterned constructions) in identifying pleonastic *it*.

Among the statistics-based methods, Bean and Riloff (1999) automatically identified existential definite NPs which are non-anaphoric. The intuition behind is that many definite NPs are not anaphoric since their meanings can be understood from general world knowledge, e.g. "the FBI". They found that existential NPs account for 63% of all definite NPs and 76% of them could be identified by syntactic or lexical means. Cherry and Bergsma (2005) extended the work of Lappin and Leass (1994) for large-scale anaphoricity determination by additionally detecting pleonastic *it*. Bergsma et al. (2008) proposed a distributional method in detecting non-anaphoric pronouns. They first extracted the surrounding context of the pronoun and gathered the distribution of words that occurred within the context from a large corpus, and then identified the pronoun either anaphoric or non-anaphoric based on the word distribution.

Among the learning-based methods, Evans (2001) automatically identified the non-anaphoricity of pronoun *it* using various kinds of lexical and syntactic features. Ng and Cardie (2002a) employed various domain-independent features in identifying anaphoric NPs. They trained an anaphoricity classifier to determine whether a NP was anaphoric or not, and employed an independently-trained coreference resolution system to only resolve those mentions which were classified as anaphoric. Experiments showed that their method improved the performance of coreference resolution by 2.0 and 2.6 to 65.8 and 64.2 in F1-measure on the MUC-6 and MUC-7 corpora, respectively. Ng (2004) examined the representation and optimization issues in computing and using anaphoricity information to improve learning-based coreference resolution. On the basis, he presented a corpus-based approach (Ng, 2009) for achieving global optimization by representing anaphoricity as a feature in coreference resolution. Experiments on the ACE 2003 corpus showed that their method improved the overall performance by 2.8, 2.2 and 4.5 to 54.5, 64.0 and 60.8 in F1-measure on the NWIRE, NPAPER and BNEWS domains, respectively. However, he did not look into the contribution of anaphoricity determination on coreference resolution of different NP types. Yang et al. (2005) made use of non-anaphors to create a special class of training instances in the twin-candidate model (Yang et al. 2003) and improved the performance by 2.9 and 1.6 to 67.3 and 67.2 in F1-measure on the MUC-6 and MUC-7 corpora, respectively. However, their experiments show that eliminating non-anaphors using an anaphoricity determination module in advance harms the performance. Denis and Balbridge (2007) employed an integer linear programming (ILP) formulation for coreference resolution which modeled anaphoricity and coreference as a joint task, such that each local model informed the other for the final assignments. Experiments on the ACE 2003 corpus showed that this joint anaphoricity-coreference ILP formulation improved the F1-measure by 3.7-5.3 on various domains. However, their experiments assume true ACE mentions (i.e. all the ACE mentions are already known from the annotated corpus). Therefore, the actual effect of this joint anaphoricity-coreference ILP formulation on fully automatic coreference resolution is still unclear. Luo (2007) proposed a twin-model for coreference resolution: a link component, which models the coreferential relationship between an anaphor and a candidate antecedent, and a creation component, which models the possibility that a NP was not coreferential with any candidate antecedent. This method combined the probabilities returned by the creation component (an anaphoricity model) with the link component (a coreference model) to score a coreference partition, such that a partition was penalized whenever an anaphoric mention was resolved. Finkel and Manning (2008) showed that transitivity constraints could be incorporated into an ILP-based coreference resolution system and much improved the performance. Zhou and Kong (2009) employed a global learning method in determining the anaphoricity of NPs via a label propagation algorithm to improve learning-based coreference resolution. Experiments on the ACE 2003 corpus demonstrated that this method was very effective. It could improve the F1-measure by 2.4, 3.1 and 4.1 on the NWIRE, NPAPER and BNEWS domains, respectively. Ng (2009) presented a novel approach to the task of anaphoricity determination based on graph minimum cuts and demonstrated the effectiveness in improving a learning-based coreference resolution system.

In summary, although anaphoricity determination plays an important role in coreference resolution and achieves certain success in improving the overall performance of coreference resolution, its contribution is still far from expectation.

2.2 Syntactic Parse Tree Structures

For a tree kernel-based method, one key problem is how to represent and capture the structured syntactic information. During recent years, various tree kernels, such as the convolution tree kernel (Collins and Duffy, 2001), the shallow parse tree kernel (Zelenko et al 2003) and the dependency tree kernel (Culota and Sorensen, 2004), have been proposed in the literature. Among these tree kernels, the convolution tree kernel represents the state-of-the art and has been successfully applied by Collins and Duffy (2002) on syntactic parsing, Zhang et al. (2006) on semantic relation extraction and Yang et al. (2006) on pronoun resolution.

Given a tree kernel, the key issue is how to generate a syntactic parse tree structure for effectively capturing the structured syntactic information. In the literature, various parse tree structures have been proposed and successfully applied in some NLP applications. As a representative, Zhang et al. (2006) investigated five parse tree structures for semantic relation extraction and found that the Shortest Path-enclosed Tree (SPT) achieves the best performance on the 7 relation types of the ACE RDC 2004 corpus. Yang et al. (2006) constructed a document-level syntactic parse tree for an entire text by attaching the parse trees of all its sentences to a new-added upper node and examined three possible parse tree structures (Min-Expansion, Simple-Expansion and Full-Expansion) that contain different substructures of the parse tree for pronoun resolution. Experiments showed that their method achieved certain success on the ACE 2003 corpus and the simple-expansion scheme performs best. However, among the three explored schemes, there exists no obvious overwhelming one, which can well cover structured syntactic information. One problem of Zhang et al. (2006)

and Yang et al. (2006) is that their parse tree structures are context-free and do not consider the information outside the sub-trees. Hence, their ability of exploring structured syntactic information is much limited. Motivated by Zhang et al. (2006) and Yang et al. (2006), Zhou et al. (2007) extended the SPT to become context-sensitive (CS-SPT) by dynamically including necessary predicate-linked path information. Zhou et al. (2008) further proposed a dynamic-expansion scheme to automatically determine a proper parse tree structure for pronoun resolution by taking predicate- and antecedent competitor-related information in consideration. Evaluation on the ACE 2003 corpus showed that the dynamic-expansion scheme can well cover necessary structured information in the parse tree for pronoun resolution. One problem with the above parse tree structures is that they may still contain unnecessary information and also miss some useful context-sensitive information. Qian et al. (2008) dynamically determined the parse tree structure for semantic relation extraction by exploiting constituent dependencies to keep the necessary information in the parse tree as well as remove the noisy information. Evaluation on the ACE RDC 2004 corpus showed that their dynamic syntactic parse tree structure outperforms all previous parse tree structures. However, their solution has the limitation in that the dependencies were found according to some manually-written ad-hoc rules and thus may not be easily applicable to new domains and applications.

This paper proposes a new scheme to dynamically determine the syntactic parse tree structure for anaphoricity determination and systematically studies the application of an explicit anaphoricity determination module in improving coreference resolution.

3 Dependency-driven Dynamic Syntactic Parse Tree

Given a full syntactic parse tree and a NP in consideration, one key issue is how to choose a proper syntactic parse tree structure to well cover structured syntactic information in the tree kernel computation. Generally, the more a syntactic parse tree structure includes, the more structured syntactic information would be available, at the expense of more noisy (or unnecessary) information.

It is well known that dependency information plays a key role in many NLP problems, such as syntactic parsing, semantic role labeling as well as semantic relation extraction. Motivated by Qian et al. (2008) and Zhou et al. (2008), we propose a new scheme to dynamically determine the syntactic parse tree structure for anaphoricity determination by exploiting constituent dependencies from both the syntactic and semantic perspectives to distinguish the necessary evidence from the unnecessary information in the syntactic parse tree. That is, constituent dependencies are explored from two aspects: syntactic dependencies and semantic dependencies.

1) **Syntactic Dependencies:** The Stanford dependency parser[1] is employed as our syntactic dependency parser to automatically extract various syntactic (i.e. grammatical) dependencies between individual words. In this paper, only immediate syntactic dependencies with current mention are considered. The intuition behind is that the immediate syntactic dependencies carry the major contextual information of current mention.
2) **Semantic Dependencies:** A state-of-the-art semantic role labeling (SRL) toolkit (Li et al. 2009) is employed for extracting various semantic dependencies related with current mention. In this paper, semantic dependencies include all the predicates heading any node in the root path from current mention to the root node and their compatible arguments (except those overlapping with current mention).

We name our parse tree structure as a dependency-driven dynamic syntactic parse tree (D-DSPT). The intuition behind is that the dependency information related with current mention in the same sentence plays a critical role in anaphoricity determination. Given the sentence enclosing the mention under consideration, we can get the D-DSPT as follows: (Figure 1 illustrates an example of the D-DSPT generation given the sentence "Mary said the woman in the room bit her" with "woman" as current mention.)

[1] http://nlp.stanford.edu/software/lex-parser.shtml

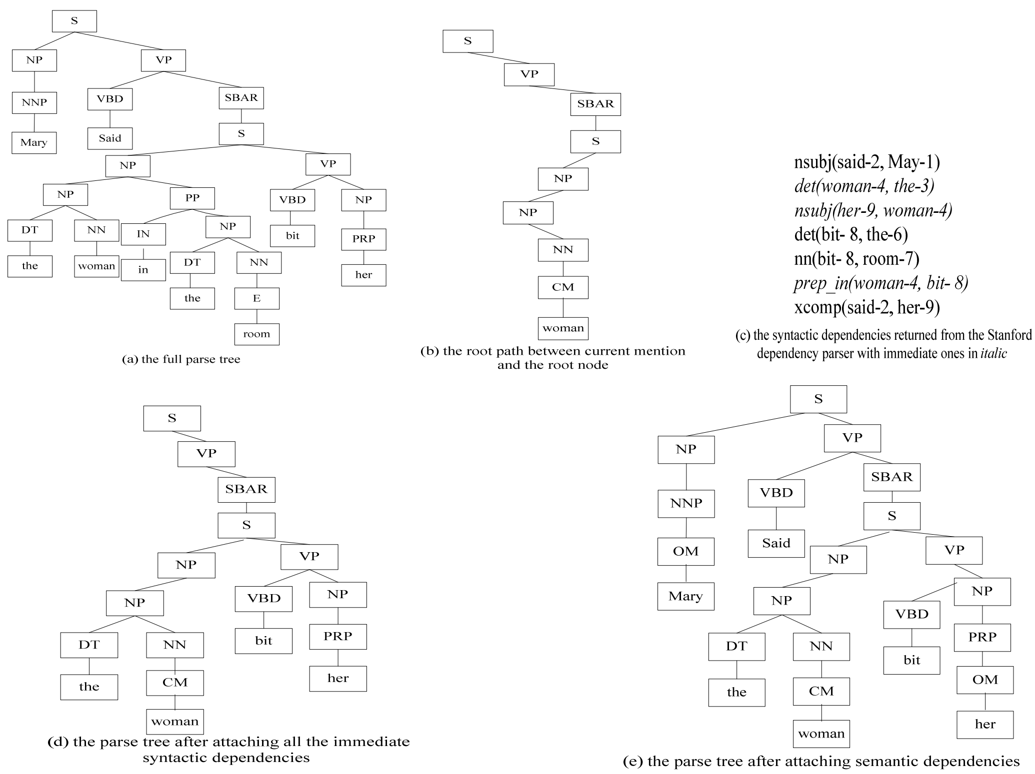


Figure 1: An example of generating the dependency-driven dynamic syntactic parse tree

1) Generating the full syntactic parse tree of the given sentence using a full syntactic parser. In this paper, the Charniak parser (Charniak 2001) is employed and Figure 1 (a) shows the resulting full parse tree.

2) Keeping only the root path from current mention to the root node of the full parse tree. Figure 1(b) shows the root path corresponding to the current mention "woman". In the following steps, we attach the above two types of dependency information to the root path.

3) Extracting all the syntactic dependencies in the sentence using a syntactic dependency parser, and attaching all the nodes, which have immediate dependency relationship with current mention, and their corresponding paths to the root path. Figure 1(c) illustrates the syntactic dependences extracted from the sentence, where the ones in italic mean immediate dependencies with current mention. Figure 1(d) shows the parse tree structure after considering syntactic dependencies.

4) Attaching all the predicates heading any node in the root path from current mention to the root node and their corresponding paths to the root path. For the example sentence, there are two predicates "said" and "bit", which head the "VP" and "S" nodes in the root path respectively. Therefore, these two predicates and their corresponding paths should be attached to the root path as shown in Figure 1(e). Note that the predicate "bit" and its corresponding path has already been attached in Stop (3). As a result, the predicate-related information can be attached. According to Zhou and Kong (2009), such information is important to definite NP resolution.

5) Extracting the semantic dependencies related with those attached predicates using a (shallow) semantic parser, and attaching all the compatible arguments (except those overlapping with current mention) and their corresponding paths to the root path. For example, as shown in Figure 1(e), since the arguments "Mary" and "her" are compatible with current mention "woman", these two nodes and their corresponding paths are attached while the argument "room" is not since its gender does not agree with current mention.

In this paper, the similarity between two parse trees is measured using a convolution tree kernel, which counts the number of common sub-tree as the syntactic structure similarity between two parse trees. For details, please refer to Collins and Duffy (2001).

4 Experimentation and Discussion

This section evaluates the performance of dependency-driven anaphoricity determination and its application in coreference resolution on the ACE 2003 corpus.

4.1 Experimental Setting

The ACE 2003 corpus contains three domains: newswire (NWIRE), newspaper (NPAPER), and broadcast news (BNEWS). For each domain, there exist two data sets, training and devtest, which are used for training and testing.

For preparation, all the documents in the corpus are preprocessed automatically using a pipeline of NLP components, including tokenization and sentence segmentation, named entity recognition, part-of-speech tagging and noun phrase chunking. Among them, named entity recognition, part-of-speech tagging and noun phrase chunking apply the same state-of-the-art HMM-based engine with error-driven learning capability (Zhou and Su, 2000 & 2002). Our statistics finds that 62.0%, 58.5% and 61.4% of entity mentions are preserved after preprocessing on the NWIRE, NPAPER and BNEWS domains of the ACE 2003 training data respectively while only 89.5%, 89.2% and 94% of entity mentions are preserved after preprocessing on the NWIRE, NPAPER and BNEWS domains of the ACE 2003 devtest data. This indicates the difficulty of coreference resolution. In addition, the corpus is parsed using the Charniak parser for syntactic parsing and the Stanford dependency parser for syntactic dependencies while corresponding semantic dependencies are extracted using a state-of-the-art semantic role labeling toolkit (Li et al. 2009). Finally, we use the SVM-light[2] toolkit with the tree kernel function as the classifier. For comparison purpose, the training parameters C (SVM) and λ (tree kernel) are set to 2.4 and 0.4 respectively, as done in Zhou and Kong (2009).

For anaphoricity determination, we report the performance in Acc^+ and Acc^-, which measure the accuracies of identifying anaphoric NPs and non-anaphoric NPs, respectively. Obviously, higher Acc^+ means that more anaphoric NPs would be identified correctly, while higher Acc^- means that more non-anaphoric NPs would be filtered out. For coreference resolution, we report the performance in terms of recall, precision, and F1-measure using the commonly-used model theoretic MUC scoring program (Vilain et al. 1995). To see whether an improvement is significant, we also conduct significance testing using paired t-test. In this paper, '***', '**' and '*' denote p-values of an improvement smaller than 0.01, in-between (0.01, 0,05] and bigger than 0.05, which mean significantly better, moderately better and slightly better, respectively.

4.2 Experimental Results

Performance of anaphoricity determination

Table 1 presents the performance of anaphoricity determination using the convolution tree kernel on D-DSPT. It shows that our method achieves the accuracies of 83.27/77.13, 86.77/80.25 and 90.02/64.24 on identifying anaphoric/non-anaphoric NPs in the NWIRE, NPAPER and BNEWS domains, respectively. This suggests that our approach can effectively filter out about 75% of non-anaphoric NPs and keep about 85% of anaphoric NPs. In comparison, in the three domains Zhou and Kong (2009) achieve the accuracies of 76.5/82.3, 78.9/81.6 and 74.3/83.2, respectively, using the tree kernel on a dynamically-extended tree (DET). This suggests that their method can filter out about 82% of non-anaphoric NPs and only keep about 76% of anaphoric NPs. In comparison, their method outperforms our method on filtering out more non-anaphoric NPs while our method outperforms their method on keeping more anaphoric NPs in coreference resolution. While a coreference resolution system can detect some non-anaphoric NPs (when failing to find the antecedent candidate), filtering out anaphoric NPs in anaphoricity determination would definitely cause errors and it is almost impossible to recover. Therefore, it is normally more important to keeping more anaphoric NPs than filtering out more non-anaphoric NPs. Table 1 further presents the performance of anaphoricity determination on different NP types. It shows that our method performs best at keeping pronominal NPs and filtering out proper NPs.

[2] http://svmlight.joachims.org/

NP Type	NWIRE		NPAPER		BNEWS	
	Acc^{+}	Acc^{-}	**Acc**$^{+}$	Acc^{-}	**Acc**$^{+}$	Acc^{-}
Pronoun	95.07	50.36	96.40	56.44	98.26	54.03
Proper NP	84.61	83.17	83.78	79.62	87.61	71.77
Definite NP	87.17	46.74	82.24	49.18	86.87	53.65
Indefinite NP	86.01	47.52	80.63	48.45	89.71	47.32
Over all	83.27	77.13	86.77	80.25	90.02	64.24

Table 1: Performance of anaphoricity determination using the D-DSPT

Performance Change	NWIRE		NPAPER		BNEWS	
	Acc^{+}	Acc^{-}	**Acc**$^{+}$	Acc^{-}	**Acc**$^{+}$	Acc^{-}
D-DSPT	83.27	77.13	86.77	80.25	90.02	64.24
-Syntactic Dependencies	78.67	72.56	80.14	73.74	87.05	60.20
-Semantic Dependencies	81.67	76.74	83.47	77.93	89.58	60.67

Table 2: Contribution of including syntactic and semantic dependencies in D-DSPT on anaphoricity determination

System		NWIRE			NPAPER			BNEWS		
		R%	P%	F	R%	P%	F	R%	P%	F
Without anaphoricity determination (Baseline)	Pronoun	70.8	57.9	63.7	76.5	63.5	69.4	70.0	60.3	64.8
	Proper NP	80.3	80.1	80.2	81.8	83.6	82.7	76.3	76.8	76.6
	Definite NP	35.9	43.4	39.2	43.1	48.5	45.6	47.9	51.9	49.8
	Indefinite NP	40.3	26.3	31.8	39.7	22.9	29.0	23.6	10.7	14.7
	Over all	***55.0***	***63.8***	***59.1***	***62.1***	***65.0***	***63.5***	***53.2***	***60.5***	***56.6***
With D-DSPT -based anaphoricity determination	Pronoun	65.9	70.2	68.0	72.6	78.7	75.5	67.7	75.8	71.5
	Proper NP	80.3	81.0	80.6	81.2	85.1	83.1	76.3	84.4	80.1
	Definite NP	32.3	63.1	42.7	38.4	61.7	47.3	42.5	66.4	51.8
	Indefinite NP	36.4	55.3	43.9	34.7	50.7	41.2	20.3	45.4	28.1
	Over all	***52.4***	***79.6***	***63.2***	***58.1***	***80.3***	***67.4***	***50.1***	***79.8***	***61.6***
With golden anaphoricity determination	Pronoun	68.6	71.5	70.1	75.2	80.4	77.7	69.1	77.8	73.5
	Proper NP	81.7	89.3	85.3	82.6	90.1	86.2	78.6	88.7	83.3
	Definite NP	41.8	85.9	56.2	44.9	85.2	58.8	45.2	87.9	59.7
	Indefinite NP	40.3	67.6	50.5	41.2	65.1	50.5	40.9	50.1	45.1
	Over all	***54.6***	***81.7***	***65.5***	***60.4***	***82.1***	***69.6***	**51.9**	**82.1**	**63.6**

Table 3: Performance of anaphoricity determination on coreference resolution

System		NWIRE			NPAPER			BNEWS		
		R%	P%	F	R%	P%	F	R%	P%	F
Zhou and Kong (2009)	Without anaphoricity determination (Baseline)	*53.1*	*67.4*	*59.4*	*57.7*	*67.0*	*62.1*	*48.0*	*65.9*	*55.5*
	With Dynamically Extended Tree-based anaphoricity determination	*51.6*	*77.2*	*61.8*	*55.2*	*78.6*	*65.2*	*47.5*	*80.3*	*59.6*
Ng (2009)	Without anaphoricity determination (Baseline)	*59.1*	*58.*	*58.6*	*60.8*	*62.6*	*61.7*	*57.7*	*52.6*	*55.0*
	With Graph Minimum Cut-based anaphoricity determination	*54.1*	*69.0*	*60.6*	*57.9*	*71.2*	*63.9*	*53.1*	*67.5*	*59.4*

Table 4: Performance comparison with other systems

Table 2 further presents the contribution of including syntactic and semantic dependencies in the D-DSPT on anaphoricity determination by excluding one or both of them. It shows that both syntactic dependencies and semantic dependencies contribute significantly (***).

Performance of coreference resolution

We have evaluated the effect of our D-DSPT-based anaphoricity determination module on coreference resolution by including it as a preprocessing step to a baseline coreference resolution system without explicit anaphoricity determination, by filtering our those non-anaphoric NPs according to the anaphoricity determination module. Here, the baseline system employs the same set of features, as adopted in the single-candidate model of Yang et al. (2003) and uses a SVM-based classifier with the feature-based RBF kernel. Table 3 presents the detailed performance of the coreference resolution system without anaphoricity determination, with D-DSPT-based anaphoricity determination and. with golden anaphoricity determination. Table 3 shows that:

1) There is a performance gap of 6.4, 6.1 and 7.0 in F1-measure on the NWIRE, NPAPER and BNEWS domain, respectively, between the coreference resolution system with golden anaphoricity determination and the baseline system without anaphoricity determination. This suggests the usefulness of proper anaphoricity determination in coreference resolution. This also agrees with Stoyanov et al. (2009) which measured the impact of golden anaphoricity determination on coreference resolution using only the annotated anaphors in both training and testing.

2) Compared to the baseline system without anaphoricity determination, the D-DSPT-based anaphoricity determination module improves the performance by 4.1(***), 3.9(***) and 5.0(***) to 63.2, 67.4 and 61.6 in F1-measure on the NWIRE, NPAPER and BNEWS domains, respectively, due to a large gain in precision and a much smaller drop in recall. In addition, D-DSPT-based anaphoricity determination can not only much improve the performance of coreference resolution on pronominal NPs (***) but also on definite NPs(***) and indefinite NPs(***) while the improvement on proper NPs can be ignored due to the fact that proper NPs can be well addressed by the simple abbreviation feature in the baseline system.

3) D-DSPT-based anaphoricity determination still lags (2.3, 2.2 and 2.0 on the NWIRE, NPAPER and BNEWS domains, respectively) behind golden anaphoricity determination in improving the overall performance of coreference resolution. This suggests that there exists some room in the performance improvement for anaphoricity determination.

Performance comparison with other systems

Table 4 compares the performance of our system with other systems. Here, Zhou and Kong (2009) use the same set of features with ours in the baseline system and a dynamically-extended tree structure in anaphoricity determination. Ng (2009) uses 33 features as described in Ng (2007) and a graph minimum cut algorithm in anaphoricity determination. It shows that the overall performance of our baseline system is almost as good as that of Zhou and Kong (2009) and a bit better than Ng's (2009).

For overall performance, our coreference resolution system with D-DSPT-based anaphoricity determination much outperforms Zhou and Kong (2009) in F1-measure by 1.4, 2.2 and 2.0 on the NWIRE, NPAPER and BNEWS domains, respectively, due to the better inclusion of dependency information. Detailed evaluation shows that such improvement comes from coreference resolution on both pronominal and definite NPs (Please refer to Table 6 in Zhou and Kong, 2009). Compared with Zhou and Kong (2009) and Ng (2009), our approach achieves the best F1-measure so far for each dataset.

5 Conclusion and Further Work

This paper systematically studies a dependency-driven dynamic syntactic parse tree (DDST) for anaphoricity determination and the application of an explicit anaphoricity determination module in improving learning-based coreference resolution. Evaluation on the ACE 2003 corpus indicates that D-DSPT-based anaphoricity determination much improves the performance of coreference resolution.

To our best knowledge, this paper is the first research which directly explores constituent dependencies in tree kernel-based anaphoricty determination from both syntactic and semantic perspectives.

For further work, we will explore more structured syntactic information in coreference resolution. In addition, we will study the interaction between anaphoricity determination and coreference resolution and better integrate anaphoricity determination with coreference resolution.

Acknowledgments

This research was supported by Projects 60873150, 60970056, and 90920004 under the National Natural Science Foundation of China, Project 200802850006 and 20093201110006 under the Specialized Research Fund for the Doctoral Program of Higher Education of China.

References

D. Bean and E. Riloff 1999. Corpus-based Identification of Non-Anaphoric Noun Phrases. ACL' 1999

S. Bergsma, D. Lin and R. Goebel 2008. Distributional Identification of Non-referential Pronouns. ACL'2008

C. Cherry and S. Bergsma. 2005. An expectation maximization approach to pronoun resolution. CoNLL'2005

M. Collins and N. Duffy. 2001. Covolution kernels for natural language. NIPS' 2001

M. Denber 1998. Automatic Resolution of Anaphoria in English. Technical Report, Eastman KodakCo.

P. Denis and J. Baldridge. 2007. Global, joint determination of anaphoricity and coreference resolution using integer programming. NAACL/HLT'2007

R. Evans 2001. Applying machine learning toward an automatic classification of it. Literary and Linguistic Computing, 16(1):45-57

F. Kong, G.D. Zhou and Q.M. Zhu. 2009 Employing the Centering Theory in Pronoun Resolution from the Semantic Perspective. EMNLP'2009

F. Kong, Y.C. Li, G.D. Zhou and Q.M. Zhu. 2009. Exploring Syntactic Features for Pronoun Resolution Using Context-Sensitive Convolution Tree Kernel. IALP'2009

S. Lappin and J. L. Herbert. 1994. An algorithm for pronominal anaphora resolution. Computational Linguistics, 20(4)

J.H. Li. G.D. Zhou, H. Zhao, Q.M. Zhu and P.D. Qian. Improving nominal SRL in Chinese language with verbal SRL information and automatic predicate recognition. EMNLP '2009

X. Luo. 2007. Coreference or not: A twin model for coreference resolution. NAACL-HLT'2007

V. Ng and C. Cardie 2002. Identify Anaphoric and Non-Anaphoric Noun Phrases to Improve Coreference Resolution. COLING'2002

V. Ng and C. Cardie 2002. Improving machine learning approaches to coreference resolution. ACL'2002

V. Ng 2004. Learning Noun Phrase Anaphoricity to Improve Coreference Resolution: Issues in Representation and Optimization. ACL' 2004

V. Ng 2009. Graph-cut based anaphoricity determination for coreference resolution. NAACL'2009

L.H. Qian, G.D. Zhou, F. Kong, Q.M. Zhu and P.D. Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. COLING'2008

W. M. Soon, H. T. Ng and D. Lim 2001. A machine learning approach to coreference resolution of noun phrase. Computational Linguistics, 27(4):521-544.

V. Stoyanov, N. Gilbert, C. Cardie and E. Riloff. 2009. Conundrums in Noun Phrase Coreference Resolution: Making Sense of the State-of-the Art. ACL'2009

B. L. Webber. 1979. A Formal Approach to Discourse Anaphora. Garland Publishing, Inc.

X.F. Yang, G.D. Zhou, J. Su and C.L. Chew. 2003. Coreference Resolution Using Competition Learning Approach. ACL'2003

X.F. Yang, J. Su and C.L. Chew. 2005. A Twin Candidate Model of Coreference Resolution with Non-Anaphor Identification Capability. IJCNLP'2005

X.F. Yang, J. Su and C.L. Chew. 2006. Kernel-based pronoun resolution with structured syntactic knowledge. COLING-ACL'2006

X.F. Yang, J. Su and C.L. Tan 2008. A Twin-Candidate Model for Learning-Based Anaphora Resolution. Computational Linguistics 34(3):327-356

M. Zhang, J. Zhang, J. Su and G.D. Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. COLING/ACL'2006

S. Zhao and R. Grishman. 2005. Extracting relations with integered information using kernel methods. ACL'2005

D. Zelenko, A. Chinatsu and R. Anthony. 2003. Kernel methods for relation extraction. Machine Learning Researching 3(2003):1083-1106

G.D. Zhou, F. Kong and Q.M. Zhu. 2008. Context-sensitive convolution tree kernel for pronoun resolution. IJCNLP'2008

G.D. Zhou and F. Kong. 2009. Global Learning of Noun Phrase Anaphoricity in Coreference Resolution via Label Propagetion. EMNLP'2009

G.D. Zhou and J. Su. 2002. Named Entity recognition using a HMM-based chunk tagger. ACL'2002

G.D. Zhou, M. Zhang, D.H. Ji and Q.M. Zhu. 2007. Tree kernel-based relation extraction with context-sensitive structured parse tree information. EMNLP/CoNLL'2007

Phrase Clustering for Smoothing TM Probabilities – or, How to Extract Paraphrases from Phrase Tables

[1]Roland Kuhn, [1]Boxing Chen, [1]George Foster and [2]Evan Stratford
[1]National Research Council of Canada
[2]University of Waterloo
[1]First.Last@nrc.gc.ca; [2]evan.stratford@gmail.com

Abstract

This paper describes how to cluster together the phrases of a phrase-based statistical machine translation (SMT) system, using information in the phrase table itself. The clustering is symmetric and recursive: it is applied both to source-language and target-language phrases, and the clustering in one language helps determine the clustering in the other. The phrase clusters have many possible uses. This paper looks at one of these uses: smoothing the conditional translation model (TM) probabilities employed by the SMT system. We incorporated phrase-cluster-derived probability estimates into a baseline loglinear feature combination that included relative frequency and lexically-weighted conditional probability estimates. In Chinese-English (C-E) and French-English (F-E) learning curve experiments, we obtained a gain over the baseline in 29 of 30 tests, with a maximum gain of 0.55 BLEU points (though most gains were fairly small). The largest gains came with medium (200-400K sentence pairs) rather than with small (less than 100K sentence pairs) amounts of training data, contrary to what one would expect from the paraphrasing literature. We have only begun to explore the original smoothing approach described here.

1 Introduction and Related Work

The source-language and target-language "phrases" employed by many statistical machine translation (SMT) systems are anomalous: they are arbitrary sequences of contiguous words extracted by complex heuristics from a bilingual corpus, satisfying no formal linguistic criteria. Nevertheless, phrase-based systems perform better than word-based systems (Koehn 2010, pp. 127-129). In this paper, we look at what happens when we cluster together these anomalous but useful entities.

Here, we apply phrase clustering to obtain better estimates for "backward" probability P(**s**|**t**) and "forward" probability P(**t**|**s**), where **s** is a source-language phrase, **t** is a target-language phrase, and phrase pair (**s**,**t**) was seen at least once in training data. The current work is thus related to work on smoothing P(**s**|**t**) and P(**t**|**s**) – see (Foster *et al.*, 2006). The relative frequency estimates for P(**s**|**t**) and P(**t**|**s**) are $P_{RF}(s \mid t) = \#(s,t)/\#t$ and $P_{RF}(t \mid s) = \#(s,t)/\#s$, where *#(s,t)* denotes the number of times phrase pair (**s**,**t**) was observed, *etc*. These estimates are typically smoothed with "lexical" estimates found by breaking phrases **s** and **t** into words. We adopt a different idea, that of smoothing P_{RF}(**s**|**t**) and P_{RF}(**t**|**s**) with estimates obtained from phrases that have similar meanings to **s** and **t**. In our experiments, the two methods were combined, yielding an improvement over lexical smoothing alone – this indicates they provide complementary information. *E.g.*, lexical estimates don't work well for non-compositional phrases like "kick the bucket" - our method might cluster this phrase with "die" and "expire" and thus provide better smoothing. The research that comes closest to ours is the work of Schwenk *et al*. (2007) on continuous space N-gram models, where a neural network is employed to smooth translation probabilities. However, both Schwenk *et al.*'s smoothing technique

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 608–616,
Beijing, August 2010

and the system to which it is applied are quite different from ours.

Phrase clustering is also somewhat related to work on paraphrases for SMT. Key papers in this area include (Bannard and Callison-Burch, 2005), which pioneered the extraction of paraphrases from bilingual parallel corpora, (Callison-Burch *et al*., 2006) which showed that paraphrase generation could improve SMT performance, (Callison-Burch, 2008) and (Zhao *et al*., 2008) which showed how to improve the quality of paraphrases, and (Marton *et al*., 2009) which derived paraphrases from monolingual data using distributional information. Paraphrases typically help SMT systems trained on under 100K sentence pairs the most.

The phrase clustering algorithm in this paper outputs groups of source-language and target-language phrases with similar meanings: paraphrases. However, previous work on paraphrases for SMT has aimed at finding translations for source-language phrases in the system's input that weren't seen during system training. Our approach is completely useless in this situation: it only generates new information for target or source phrases that are already in the system's phrase table. Thus, we find paraphrases for many of the source and target phrases that **are** in the phrase table, while the work cited above looks for paraphrases of source phrases that are **not** in the phrase table.

Our work also differs from most work on paraphrases in that information is extracted not from sources outside the SMT system (*e.g.*, pivot languages or thesauri) but from the system's phrase table. In this respect if no other, it is similar to Chiang's classic work on hierarchical phrase-based systems (Chiang, 2005), though Chiang was mining a very different type of information from phrase tables.

Because of all these differences between work on paraphrasing and the phrase clustering approach, both in terms of the input information and where they are best applied, we did not experimentally compare the two approaches.

2 Deriving Conditional Probabilities from Phrase Clusters

Given phrase clusters in the source and target languages, how would one derive estimates for conditional probabilities P(**s**|**t**) and P(**t**|**s**)? We assume that the clustering is "hard": each source phrase **s** belongs to exactly one cluster C(**s**), and each target phrase **t** belongs to exactly one cluster C(**t**). Some of these clusters will contain singleton phrases, and others will contain more than one phrase. Let "#" denote the total number of observations in the training data associated with a phrase or phrase cluster. *E.g.*, suppose the English cluster C_S contains the three phrases "red", "dark red", and "burgundy", with 50, 25, and 10 observations in the training data respectively – then $\#(C_S) = 85$. Also, let $\#(C_S,C_T)$ be the number of co-occurrences in the training data of source-language cluster C_S and target-language cluster C_T.

The phrase-cluster-based probabilities P_{PC} are:

$$P_{PC}(s \mid t) = P(s \mid C(s)) \times P(C(s) \mid C(t)) = \frac{\#(s)}{\#C(s)} \times \frac{\#(C(s),C(t))}{\#C(t)} \quad (1)$$

and

$$P_{PC}(t \mid s) = P(t \mid C(t)) \times P(C(t) \mid C(s)) = \frac{\#(t)}{\#C(t)} \times \frac{\#(C(s),C(t))}{\#C(s)} \quad (2)$$

Note that the P_{PC} will often be non-zero where the corresponding relative frequency estimates P_{RF} were zero (the opposite can't happen). Also, the P_{PC} will be most useful where the phrase being conditioned on was seldom seen in the training data. If **t** was seen 1,000 times during training, the P_{RF}(**s**|**t**) are reliable and don't need smoothing; but if **t** was seen 6 times, P_{PC}(**s**|**t**) may yield valuable extra information. The same kind of argument applies to estimation of P(**t**|**s**).

3 Clustering Phrases

We used only information "native" to phrase tables to cluster phrases. Two types of similarity metric between phrases or phrase clusters were employed: <u>count-based metrics</u> and <u>edit-based metrics</u>. The former are based on phrase co-occurrence counts; the latter are based on the word sequences that make up the phrases. Each has its advantages. Count-based metrics can deduce from the similar translations of two phrases that they have similar meanings, despite dissimilarity between the two word sequences – *e.g.*, they can deduce that "red" and "burgundy" belong in the same cluster. However, these metrics are unreliable when total counts are low, since phrase co-occurrences are determined by a noisy alignment process. Edit-based metrics are independent of how often phrases were observed. However, sometimes they can be fooled by phrases that have similar word sequences but different meanings (*e.g.*, "the dog bit the man"

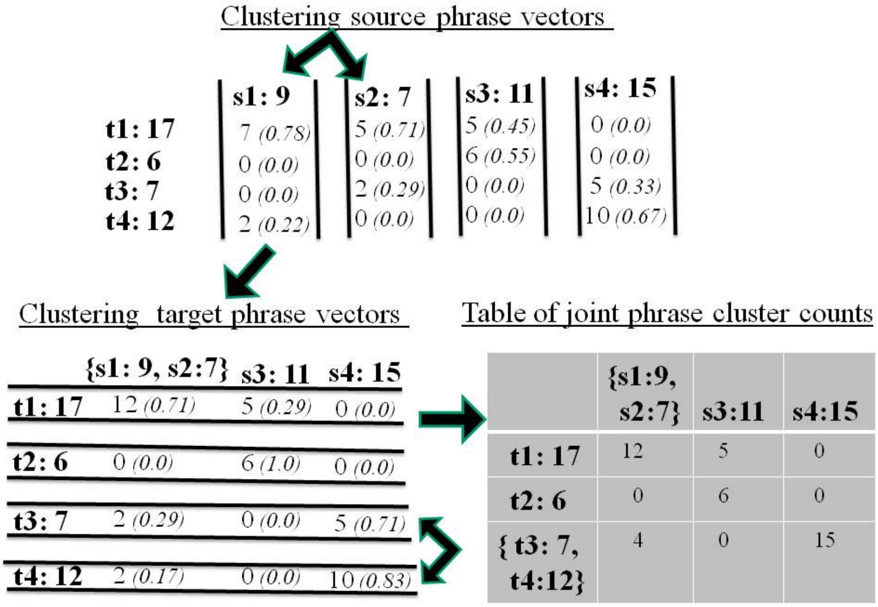


Figure 1: Example of phrase clustering

and "the man bit the dog", or "walk on the beach" and "don't walk on the beach"). In our experiments, we used a combination of count-based and edit-based metrics to cluster phrases (by simply multiplying the metrics together). However, we invested most of our effort in perfecting the count-based component: our edit-based metric was fairly naïve.

If we rely mainly on count-based similarity between phrases to cluster them, and this kind of similarity is most reliable when phrases have high counts, yet we need phrase-cluster-based estimates most for phrases with low counts, aren't we carrying out clustering on the phrases that need it least? Our hope was that there is a class of phrases with intermediate counts (*e.g.*, with 3-15 observations in the training data) that can be clustered reliably, but still benefit from phrase-cluster-based probability estimates.

3.1 Count-based clustering: overview

Figure 1 shows count-based phrase clustering. One first arbitrarily picks a language (either source or target) and then clusters together some of the phrases in that language. One then switches to the other language and clusters phrases in that language, then switches back to the first one, and so on until enough clustering has taken place.

Each phrase or phrase cluster is represented by the vector of its co-occurrence counts. To calculate the similarity between two phrase clusters, one first normalizes their count vectors. At the top of **Figure 1**, source phrase **s1** occurred 9 times: 7 times aligned with target phrase **t1**, 2 times aligned with **t4**. For source similarity computation, the entry for (**s1**,**t1**) is normalized to 7/9 = 0.78 and the entry for (**s1**,**t4**) is normalized to 2/9 = 0.22 (these normalized values are shown in brackets and italics after the counts).

The two most similar normalized vectors at the top of **Figure 1** are those associated with phrases **s1** and **s2**. These phrases are merged by adding corresponding counts, yielding a new vector associated with the new phrase cluster {**s1**, **s2**}. In real life, one would now do more source-language clustering on the source language side; in this example, we immediately proceed to target-language clustering (carried out in target language space). Note that the target similarity calculations are affected by the previous source clustering (because **s1** and **s2** are now represented by the same coordinate, **t3** and **t4** are now closer than they were in the initial table). In this manner, we can iterate back and forth between the two languages. The final output is a table of joint phrase cluster counts, which is used to estimate the P_{PC} (see previous section).

3.2 Count-based clustering: details

Count-based similarity is computed as follows:

1. Phrase alignment is a noisy process, so we first apply a transformation analogous to *tf-idf* in information retrieval (Salton and McGill, 1986) to phrase cluster

counts. For source similarity computation, each co-occurrence count $\#(C_S,C_T)$ between source cluster C_S and target cluster C_T is multiplied by a factor that reflects the information content of C_T. Let $\#diff(C_S)$ be number of clusters on the source side, and let $\#[C_T>0]$ for a particular target cluster C_T be the number of source clusters C_S that co-occur with C_T. Then let $\#'(C_S,C_T) = \#(C_S,C_T) \times \log(\#diff(C_S)/\#[C_T>0])$. Similarly, for target similarity computation, let $\#'(C_S,C_T) = \#(C_S,C_T) \times \log(\#diff(C_T)/\#[C_S>0])$. *E.g.*, in source similarity computation, if C_T co-occurs with all source clusters, its contribution will be set to zero (because it carries little information).

2. We normalize by dividing each vector of *tf-idf* counts $\#'(C_S,C_T)$ by the total number of observations in the vector.
3. We compute the similarity between each pair of *tf-idf* vectors using either the cosine measure (Salton and McGill, 1986) or one of a family of probabilistic metrics described below.
4. We cluster together the most similar vectors; this involves summing the unmodified counts $\#(C_S,C_T)$ of the vectors (*i.e.*, the *tf-idf* transformation is only applied for the purposes of similarity calculation and is not retained).

Now, we'll describe the probabilistic metrics we considered. For a count vector of dimension D, $\underline{\mathbf{u}} = (u_1, u_2, \ldots, u_D)$, define a function $I(\underline{\mathbf{u}}) = u_1 \times \log(u_1/\sum_i u_i) + \ldots + u_D \times \log(u_D/\sum_i u_i)$. $I(\underline{\mathbf{u}})$ is a measure of how well the data in $\underline{\mathbf{u}}$ are modeled by the normalized vector $(u_1/\Sigma_i u_i, \ldots, u_D/\Sigma_i u_i)$. Thus, when two count vectors $\underline{\mathbf{u}}$ and $\underline{\mathbf{v}}$ are merged (by adding them) we have the following measure of the loss in modeling accuracy:

<u>Probability Loss (PL):</u>

$$PL(\underline{\mathbf{u}},\underline{\mathbf{v}}) = I(\underline{\mathbf{u}}) + I(\underline{\mathbf{v}}) - I(\underline{\mathbf{u}}+\underline{\mathbf{v}}). \quad (3)$$

However, if we choose merges with the lowest PL, we will usually merge only vectors with small counts. We are more interested in the average impact of a merge, so we define

<u>Average Probability Loss (APL):</u>

$$APL(\underline{\mathbf{u}},\underline{\mathbf{v}}) = (I(\underline{\mathbf{u}}) + I(\underline{\mathbf{v}}) - I(\underline{\mathbf{u}}+\underline{\mathbf{v}}))/(\sum_i u_i + \sum_i v_i). \quad (4)$$

In our initial experiments, APL worked better than PL. However, APL had a strange side-effect. Most of the phrase clusters it induced made intuitive sense, but there were typically three or four clusters with large numbers of observations on both language sides that grouped together phrases with wildly disparate meanings. Why does APL induce these "monster clusters"?

Consider two count vectors $\underline{\mathbf{u}}$ and $\underline{\mathbf{v}}$. If $\Sigma_i u_i$ is very big and $\Sigma_i v_i$ is small, then $I(\underline{\mathbf{u}})$ and $I(\underline{\mathbf{u}} + \underline{\mathbf{v}})$ will be very similar, and APL will be approximately $I(\underline{\mathbf{v}}) / [\Sigma_i u_i + \Sigma_i v_i]$ which will be close to zero. Thus, the decision will probably be made to merge $\underline{\mathbf{u}}$ and $\underline{\mathbf{v}}$, even if they have quite different semantics. The resulting cluster, whose counts are represented by $\underline{\mathbf{u}} + \underline{\mathbf{v}}$, is now even bigger and even more likely to swallow up other small count vectors in the next rounds of merging: it becomes a kind of black hole.

To deal with this problem, we devised another metric. Let $I(\underline{\mathbf{u}} \mid \underline{\mathbf{v}}) = u_1 \times \log(v_1/\sum_i v_i) + \ldots + u_D \times \log(v_D/\sum_i v_i)$. This is a measure of how well the counts in $\underline{\mathbf{v}}$ predict the distribution of counts in $\underline{\mathbf{u}}$. Then let

<u>Maximum Average Probability Loss (MAPL):</u>

$$MAPL(\underline{\mathbf{u}},\underline{\mathbf{v}}) = \max\left(\frac{I(\underline{\mathbf{u}}) - I(\underline{\mathbf{u}} \mid \underline{\mathbf{u}}+\underline{\mathbf{v}})}{\sum_i u_i}, \frac{I(\underline{\mathbf{v}}) - I(\underline{\mathbf{v}} \mid \underline{\mathbf{u}}+\underline{\mathbf{v}})}{\sum_i v_i}\right) \quad (5)$$

The first term inside the maximum indicates the average probability loss for an observation in $\underline{\mathbf{u}}$ when it is modeled by $\underline{\mathbf{u}}+\underline{\mathbf{v}}$ instead of $\underline{\mathbf{u}}$; similarly, the second term indicates the average probability loss for an observation in $\underline{\mathbf{v}}$. If we merge vector pairs with the lowest values of MAPL, we will never merge vectors in a way that will cause a large loss to either of the two parents.

In practice, we found that all these metrics worked better when multiplied by the Dice coefficient based distance. For $\underline{\mathbf{u}}$ and $\underline{\mathbf{v}}$, this is $Dice(\underline{\mathbf{u}},\underline{\mathbf{v}}) = 1 - \frac{2 \times |\underline{\mathbf{u}} \cap \underline{\mathbf{v}}|}{|\underline{\mathbf{u}}| + |\underline{\mathbf{v}}|}$, where "$|\underline{\mathbf{u}}|$" means the number of non-zero count entries in $\underline{\mathbf{u}}$, and "$|\underline{\mathbf{u}} \cap \underline{\mathbf{v}}|$" is the number of count entries that are non-zero in $\underline{\mathbf{u}}$ and $\underline{\mathbf{v}}$.

3.3 Edit-based similarity

In most of our experiments, count-based metrics were combined with edit-based metrics; we put little effort into optimizing the edit metrics. Let MCWS stand for "maximum common word sequence". For phrases p_1 and p_2, we define

$$Edit(p_1, p_2) = 1 - \frac{2 \times len(MCWS(p_1, p_2))}{len(p_1) + len(p_2)}. \quad (6)$$

where *len*() returns the number of words. This metric doesn't take word identities into account; in future work, we may weight differences involving content words more heavily.

We also defined an edit-based metric for distance between phrase clusters. Let cluster 1 have phrases "red" (10); "burgundy" (5); "resembling scarlet" (2) and cluster 2 have "dark burgundy" (7); "scarlet" (3) (number of observations in brackets). What is the edit distance between clusters 1 and 2? We defined the distance as that between the two phrases with the most observations in each cluster. Thus, distance between clusters 1 and 2 would be Edit("red", "dark burgundy")=1.0. Other definitions are possible.

3.4 Examples of phrase clusters

Figure 2 shows an English phrase cluster learned during C-E experiments by a metric combining count-based and edit-based information. Each phrase is followed by its count in brackets; we don't show phrases with low counts. Since our edit distance sees words as atoms (it doesn't know about morphology), the phrases containing "emancipating" were clustered with phrases containing "emancipation" based on count information, rather than because of the common stem.

Figure 3 shows part of a French phrase cluster learned during F-E experiments by the same mixed metric. The surface forms are quite varied, but most of the phrases mean "to assure or to guarantee that something will happen". An interesting exception is "pas faire" – it means not to do something ("pas" is negative). This illustrates why we need a better edit distance that heavily weights negative words.

```
emancipating (247), emancipate
(167), emancipate our (73), emanci-
pating thinking (67), emancipate
our minds (46), further emancipate
(45), emancipate the (38), emanci-
pate the mind (38), emancipating
minds (33), emancipate their (32),
emancipate their minds (27), eman-
cipating our minds (24), emancipat-
ing our (21), emancipate our mind
(21), further emancipate our (19),
emancipate our thinking (14), fur-
ther emancipate their (11), emanci-
pating the minds (9), emancipate
thinking (8), unfettering (8) ...
```

Figure 2: partial English phrase cluster

```
garantir que (64), assurer que
(46), veiller à ce que (27), afin
de garantir (24), faire en sorte
(19), de garantir que (16), afin de
garantir que (14), faire des (14),
de veiller à ce (14), s' assurer
que (13), de veiller à ce que (13),
pour garantir que (13), de faire en
sorte (8), de faire en sorte que
(7), à garantir que (6), pas faire
(5), de veiller (5)…
```

Figure 3: partial French phrase cluster

4 Experiments

We carried out experiments on a standard one-pass phrase-based SMT system with a phrase table derived from merged counts of symmetrized IBM2 and HMM alignments; the system has both lexicalized and distance-based distortion components (there is a 7-word distortion limit) and employs cube pruning (Huang and Chiang, 2007). The baseline is a loglinear feature combination that includes language models, the distortion components, relative frequency estimators $P_{RF}(\mathbf{s}|\mathbf{t})$ and $P_{RF}(\mathbf{t}|\mathbf{s})$ and lexical weight estimators $P_{LW}(\mathbf{s}|\mathbf{t})$ and $P_{LW}(\mathbf{t}|\mathbf{s})$. The $P_{LW}()$ components are based on (Zens and Ney, 2004); Foster *et al.* (2006) found this to be the most effective lexical smoothing technique. The phrase-cluster-based components $P_{PC}(\mathbf{s}|\mathbf{t})$ and $P_{PC}(\mathbf{t}|\mathbf{s})$ are incorporated as additional loglinear feature functions. Weights on feature functions are found by lattice MERT (Macherey *et al.*, 2008).

4.1 Data

We evaluated our method on C-E and F-E tasks. For each pair, we carried out experiments on training corpora of different sizes. C-E data were from the NIST[1] 2009 evaluation; all the allowed bilingual corpora except the *UN corpus*, *Hong Kong Hansard and Hong Kong Law corpus* were used to estimate the translation model. For C-E, we trained two 5-gram language models: the first on the English side of the parallel data, and the second on the English *Gigaword corpus*.

Our C-E development set is made up mainly of data from the NIST 2005 test set; it also includes some balanced-genre web-text from the NIST training material. Evaluation was performed on the NIST 2006 and 2008 test sets. **Table 1** gives figures for training, development and test corpora for C-E tasks; |S| is the number of sentences, and |W| is the number of words. There are four references for dev and test sets.

[1] http://www.nist.gov/speech/tests/mt

			Chi	Eng
All parallel Train		|S|	3.3M	
		|W|	68.2M	66.5M
Dev		|S|	1,506	1,506×4
Test	NIST06	|S|	1,664	1,664×4
	NIST08	|S|	1,357	1,357×4
Gigaword		|S|	-	11.7M

Table 1: Statistics for Chinese-to-English tasks.

			Fre	Eng
Train	Europarl	|S|	1.6M	
		|W|	51.3M	46.6M
Dev	2008	|S|	2,051	
Test	2009	|S|	2,525	
	2010	|S|	2,489	
GigaFrEn		|S|	-	22.5M

Table 2: Statistics for French-to-English tasks.

Lang (#sent)		C-E (3.3M)		F-E (1.6M)	
		#count-1	#other	#count-1	#other
Src	Before clustering	11.3M	5.7M	28.1M	21.2M
	After clustering	11.3M	5.3M	28.1M	19.3M
	#clustered	0	0.4M	0	1.9M
Tgt	Before clustering	11.9M	6.0M	25.6M	20.4M
	After clustering	11.9M	5.6M	25.6M	18.5M
	#clustered	0	0.4M	0	1.9M

Table 3: # phrase classes before & after clustering.

For F-E tasks, we used WMT 2010[2] F-E track data sets. Parallel *Europarl* data are used for training; WMT Newstest 2008 set is the dev set, and WMT Newstest 2009 and 2010 are the test sets. One reference is provided for each source input sentence. Two language models are used in this task: one is the English side of the parallel data, and the second is the English side of the *GigaFrEn* corpus. **Table 2** summarizes the training, development and test corpora for F-E tasks.

4.2 Amount of clustering and metric

For both C-E and E-F, we assumed that phrases seen only once in training data couldn't be clustered reliably, so we prevented these "count 1" phrases from participating in clustering. The key clustering parameter is the number of merge operations per iteration, given as a percentage of the number of potential same-language phrase pairs satisfying a simple criterion (some overlap in translations to the other language). Preliminary tests involving the FBIS corpus (about 8% of the C-E data) caused us to set this parameter at 5%. For C-E, we first clustered Chinese with this 5% value, then English with the same amount. For F-E, we first clustered French, then English, using 5% in both cases.

Table 3 shows the results. Only 2-4% of the total phrases in each language end up in a cluster (that's 6.5-9% of eligible phrases, *i.e.*, of phrases that aren't "count 1"). However, about 20-25% of translation probabilities are smoothed for both language pairs. Based on these preliminary tests, we decided to use $Edit \times Dice \times MAPL$ ($Edit \times DMAPL$) as our metric (though $Edit \times Cosine$ was a close runner-up).

4.3 Results and discussion

Our evaluation metric is IBM BLEU (Papineni *et al.*, 2002), which performs case-insensitive matching of n-grams up to $n = 4$. Our first experiment evaluated the effects of the phrase clustering features given various amounts of training data. **Figure 4** gives the BLEU score improvements for the two language pairs, with results for each pair averaged over two test sets (training data size shown as #sentences). The improvement is largest for medium amounts of training data. Since the F-E training data has more words per sentence than C-E, the two peaks would have been closer together if we'd put #words on the x axis: improvements for both tasks peak around 6-8 M English words. For more details, refer to **Table 4** and **Table 5**. The biggest improvement is 0.55 BLEU for the NIST06 test. More importantly, cluster features yield gains in 29 of 30 experiments. Surprisingly, a reviewer asked if we'd done significance tests on the individual results shown in **Tables 4** and **5.** Most likely, many of these individual results are insignificant, but so what? Based on the tables, the probability of the null hypothesis that our method has no effect is equivalent to that of tossing a fair coin 30 times and getting 29 heads (if we adopt an independence approximation).

In the research on paraphrases cited earlier, paraphrases tend to be most helpful for <u>small</u> amounts of training data. By contrast, our approach seems to be most helpful for <u>medium</u> amounts of training data (200-400K sentence

[2] http://www.statmt.org/wmt10/

Data size	Nist06			Nist08		
	Baseline	+phrase-clustering	Improv.	Baseline	+phrase-clustering	Improv.
25K	21.66	21.88	0.22	15.80	15.99	0.19
50K	23.23	23.43	0.20	17.69	17.84	0.15
100K	25.83	26.24	0.41	20.08	20.27	0.19
200K	27.80	28.26	0.46	21.28	21.58	0.30
400K	29.61	30.16	**0.55**	23.37	23.75	**0.38**
800K	30.87	31.17	0.30	24.41	24.65	0.24
1.6M	32.94	33.10	0.16	25.61	25.72	0.11
3.3M	33.59	33.64	0.05	26.84	26.85	0.01

Table 4: BLEU(%) scores for C-E with the various training corpora, including baseline results, results for with phrase clustering, and the absolute improvements. Corpus size is measured in sentences.

Data size	Newstest2009			Newstest2010		
	Baseline	+phrase-clustering	Improv.	Baseline	+phrase-clustering	Improv.
25K	20.21	20.37	0.16	20.54	20.73	0.19
50K	21.25	21.44	0.19	21.95	22.11	0.16
100K	22.56	22.86	0.30	23.44	23.69	0.25
200K	23.67	24.02	**0.35**	24.31	24.71	**0.40**
400K	24.36	24.50	0.14	25.28	25.46	0.18
800K	24.92	24.97	0.05	25.80	25.90	0.10
1.6M	25.47	25.47	0.00	26.35	26.37	0.02

Table 5: BLEU(%) scores for F-E with the various training corpora, including baseline results without phrase clustering feature, results for phrase clustering, and the absolute improvements.

pairs). We attribute this to the properties of count-based clustering. When there is little training data, clustering is unreliable; when there is much data, clustering is reliable but unneeded, because most relative frequencies are well-estimated. In between, phrase cluster probability estimates are both reliable and useful.

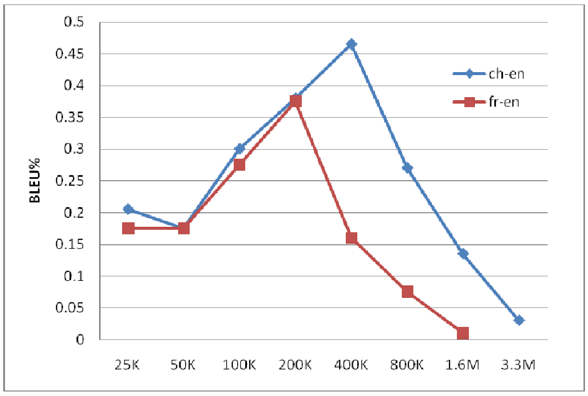


Figure 4: Average BLEU improvement for C-E and F-E tasks (each averaged over two tests) *vs.* #training sent.

Finally, we carried out experiments to see if some of our earlier decisions were correct. Were we right to use DMAPL instead of cosine as the count-based component of our metric? Experiments with $Edit \times DMAPL$ *vs.* $Edit \times Cosine$ on 400K C-E (tested on NIST06 and NIST08) and on 200K F-E (tested on Newstest2009 and 2010) showed a tiny advantage for $Edit \times DMAPL$ of about 0.06 BLEU. So we probably didn't make the wrong decision here (though it doesn't matter much). Were we right to include the *Edit* component? Carrying out analogous experiments with $Edit \times DMAPL$ *vs.* *DMAPL*, we found that dropping *Edit* caused a loss of 0.1-0.2 BLEU for all four test sets. Here again, we made the right decision.

In a final experiment, we allowed "count 1" phrases to participate in clustering (using $Edit \times DMAPL$). The resulting C-E system had somewhat more clustered phrases than the previous one (for both Chinese and English, about 3.5% of phrases were in clusters compared to 2.5% in the previous system). To our surprise, this led to a slight improvement in BLEU: the 400K C-E system now yielded 30.25 on NIST06 (up 0.09) and 23.88 on NIST08 (up 0.13). The F-E system where "count 1" clustering is allowed also had more phrases in clusters than the system where it's prohibited (the former has just under 10% of French and English phrases in clusters *vs.*

4% for the latter). For F-E, the 200K system allowing "count 1" clustering again yielded a slightly higher BLEU: 24.07 on Newstest2009 and 24.76 on Newstest2010 (up 0.05 in both cases). Thus, our decision not to allow "count 1" phrases to participate in clustering in the Table 4 and 5 experiments appears to have been a mistake. We suspect we can greatly improve handling of "count 1" phrases – *e.g.*, by weighting the Edit component of the similarity metric more heavily when assigning these phrases to clusters.

5 Conclusion and Future Work

We have shown that source-language and target-language phrases in the phrase table can be clustered, and that these clusters can be used to smooth "forward" and "backward" estimates P(**t**|**s**) and P(**s**|**t**), yielding modest but consistent BLEU gains over a baseline that included lexical smoothing. Though our experiments were done on a phrase-based system, this method could also be applied to hierarchical phrase-based SMT and syntactic SMT systems. There are several possibilities for future work based on new applications for phrase clusters:

- In the experiments above, we used phrase clusters to smooth P(t|s) and P(s|t) when the pair (s,t) was observed in training data. However, the phrase clusters often give non-zero probabilities for P(t|s) and P(s|t) when s and t were both in the training data, but didn't co-occur. We could allow the decoder to consider such "invented" phrase pairs (s,t).
- Phrase clusters could be used to construct target language models (LMs) in which the basic unit is a phrase cluster rather than a word. For instance, a tri-cluster model would estimate the probability of phrase p at time i as a function of its phrase cluster, Ci(p), and the two preceding phrase clusters Ci-1 and Ci-2: $P(\mathbf{p}) = f(\mathbf{p} \mid C_i(\mathbf{p})) \times f(C_i \mid C_{i-1}C_{i-2})$.
- Lexicalized distortion models could be modified so as to condition distortion events on phrase clusters.
- We could build SMT grammars in which the terminals are phrases and the parents of terminals are phrase clusters.

The phrase clustering algorithm described above could be improved in several ways:

- In the above, the edit distance between phrases and between phrase clusters was crudely defined. If we improve edit distance, it will have an especially large impact on "count 1" phrases, for which count-based metrics are unreliable and which are a large proportion of all phrases. The edit distance between two phrases weighted all words equally: preferably, weights for word substitution, insertion, or deletion would be learned from purely count-derived phrase clusters (content words and negative words might have heavier weights than other words). The edit distance between two phrase clusters was defined as the edit distance between the phrases with the most observations in each cluster. E.g., distance to the phrase cluster in Figure 2 is defined as the phrase edit distance to "emancipating". Instead, one could allow a cluster to be characterized by (e.g.) up to three phrases, and let distance between two clusters be the minimum or average pairwise edit distance between these characteristic phrases.
- To cluster phrases, we only used information derived from phrase tables. In future, we could also use the kind of information used in work on paraphrases, such as the context surrounding phrases in monolingual corpora, entries in thesauri, and information from pivot languages.
- The phrase clustering above was "hard": each phrase in either language belongs to exactly one cluster. We could modify our algorithms to carry out "soft" clustering. For instance, we could interpolate the probabilities associated with a phrase with probabilities from its neighbours.
- Clustering is a primitive way of finding latent structure in the table of joint phrase counts. One could apply principal component analysis or a related algorithm to this table.

References

C. Bannard and C. Callison-Burch. "Paraphrasing with Bilingual Parallel Corpora". *Proc. ACL*, pp. 597-604, Ann Arbor, USA, June 2005.

C. Callison-Burch, P. Koehn, and M. Osborne. "Improved Statistical Machine Translation Using Paraphrases". *Proc. HLT/NAACL*, pp. 17-24, New York City, USA, June 2006.

C. Callison-Burch. "Syntactic Constraints on Paraphrases Extracted from Parallel Corpora". *Proc. EMNLP*, pp. 196-205, Honolulu, USA, October 2008.

D. Chiang. "A hierarchical phrase-based model for statistical machine translation". *Proc. ACL*, pp. 263-270, Ann Arbor, USA, June 2005.

G. Foster, R. Kuhn, and H. Johnson. "Phrasetable smoothing for statistical machine translation". *Proc. EMNLP*, pp. 53-61, Sydney, Australia, July 2006.

L. Huang and D. Chiang. "Forest Rescoring: Faster Decoding with Integrated Language Models". *Proc. ACL*, pp. 144-151, Prague, Czech Republic, June 2007.

P. Koehn. 2010. *Statistical Machine Translation*. Cambridge University Press, Cambridge, UK.

W. Macherey, F. Och, I. Thayer, and J. Uszkoreit. "Lattice-based Minimum Error Rate Training for Statistical Machine Translation". *Proc. EMNLP*, pp. 725-734, Honolulu, USA, October 2008.

Y. Marton, C. Callison-Burch, and Philip Resnik. "Improved Statistical Machine Translation Using Monolingually-Derived Paraphrases". *Proc. EMNLP*, pp. 381-390, Singapore, August 2009.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. "Bleu: a method for automatic evaluation of machine translation". *Proc. ACL*, pp. 311–318, Philadelphia, July 2002.

G. Salton and M. McGill. 1986. *Introduction to Modern Information Retrieval.* McGraw-Hill Inc., New York, USA.

H. Schwenk, M. Costa-jussà, and J. Fonollosa. "Smooth Bilingual N-gram Translation". *Proc. Joint EMNLP/CoNLL*, pp. 430-438, Prague, Czech Republic, June 2007.

R. Zens and H. Ney. "Improvements in phrase-based statistical machine translation". *Proc. ACL/HLT*, pp. 257-264, Boston, USA, May 2004.

S. Zhao, H. Wang, T. Liu, and S. Li. "Pivot Approach for Extracting Paraphrase Patterns from Bilingual Corpora". *Proc. ACL/HLT*, pp. 780-788, Columbus, USA, June 2008.

Revisiting Context-based Projection Methods for Term-Translation Spotting in Comparable Corpora

Audrey Laroche
OLST – Dép. de linguistique et de traduction
Université de Montréal
audrey.laroche@umontreal.ca

Philippe Langlais
RALI – DIRO
Université de Montréal
felipe@iro.umontreal.ca

Abstract

Context-based projection methods for identifying the translation of terms in comparable corpora has attracted a lot of attention in the community, *e.g.* (Fung, 1998; Rapp, 1999). Surprisingly, none of those works have systematically investigated the impact of the many parameters controlling their approach. The present study aims at doing just this. As a test-case, we address the task of translating terms of the medical domain by exploiting pages mined from Wikipedia. One interesting outcome of this study is that significant gains can be obtained by using an association measure that is rarely used in practice.

1 Introduction

Identifying translations of terms in comparable corpora is a challenge that has attracted many researchers. A popular idea that emerged for solving this problem is based on the assumption that the context of a term and its translation share similarities that can be used to rank translation candidates (Fung, 1998; Rapp, 1999). Many variants of this idea have been implemented.

While a few studies have investigated pattern matching approaches to compare source and target contexts (Fung, 1995; Diab and Finch, 2000; Yu and Tsujii, 2009), most variants make use of a bilingual lexicon in order to translate the words of the context of a term (often called *seed words*). Déjean et al. (2005) instead use a bilingual thesaurus for translating these.

Another distinction between approaches lies in the way the context is defined. The most common practice, the so-called window-based approach, defines the context words as those cooccuring significantly with the source term within windows centered around the term.[1] Some studies have reported gains by considering syntactically motivated co-occurrences. Yu and Tsujii (2009) propose a resource-intensive strategy which requires both source and target dependency parsers, while Otero (2007) investigates a lighter approach where a few hand coded regular expressions based on POS tags simulate source parsing. The latter approach only requires a POS tagger of the source and the target languages as well as a small parallel corpus in order to project the source regular expressions.

Naturally, studies differ in the way each cooccurrence (either window or syntax-based) is weighted, and a plethora of association scores have been investigated and compared, the likelihood score (Dunning, 1993) being among the most popular. Also, different similarity measures have been proposed for ranking target context vectors, among which the popular cosine measure.

The goal of the different authors who investigate context-projection approaches also varies. Some studies are tackling the problem of identifying the translation of general words (Rapp, 1999; Otero, 2007; Yu and Tsujii, 2009) while others are addressing the translation of domain specific terms. Among the latter, many are translating single-word terms (Chiao and Zweigenbaum, 2002; Déjean et al., 2005; Prochasson et

[1] A stoplist is typically used in order to prevent function words from populating the context vectors.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 617–625,
Beijing, August 2010

al., 2009), while others are tackling the translation of multi-word terms (Daille and Morin, 2005). The type of discourse might as well be of concern in some of the studies dedicated to bilingual terminology mining. For instance, Morin et al. (2007) distinguish popular science versus scientific terms, while Saralegi et al. (2008) target popular science terms only.

The present discussion only focuses on a few number of representative studies. Still, it is already striking that a direct comparison of them is difficult, if not impossible. Differences in resources being used (in quantities, in domains, etc.), in technical choices made (similarity measures, context vector computation, etc.) and in objectives (general versus terminological dictionary extraction) prevent one from establishing a clear landscape of the various approaches.

Indeed, many studies provide some figures that help to appreciate the influence of some parameters in a given experimental setting. Notably, Otero (2008) studies no less than 7 similarity measures for ranking context vectors while comparing window and syntax-based methods. Morin et al. (2007) consider both the log-likelihood and the mutual information association scores as well as the Jaccard and the cosine similarity measures.

Ideally, a benchmark on which researchers could run their translation finder would ease the comparison of the different approaches. However, designing such a benchmark that would satisfy the evaluation purposes of all the researchers is far too ambitious a goal for this contribution. Instead, we investigate the impact of some major factors influencing projection-based approaches on a task of translating 5,000 terms of the medical domain (the most studied domain), making use of French and English Wikipedia pages extracted monolingually thanks to an information retrieval engine. While the present work does not investigate all the parameters that could potentially impact results, we believe it constitutes the most complete and systematic comparison made so far with variants of the context-based projection approach.

In the remainder of this paper, we describe the projection-based approach to translation spotting in Section 2 and detail the parameters that directly influence its performance. The experimental protocol we followed is described in Section 3 and we analyze our results in Section 4. We discuss the main results in the light of previous work and propose some future avenues in Section 5.

2 Projection-based variants

The approach we investigate for identifying term translations in comparable corpora is similar to (Rapp, 1999) and many others. We describe in the following the different steps it encompasses and the parameters we are considering in the light of typical choices made in the literature.

2.1 Approach

Step 1 A comparable corpus is constructed for each term to translate. In this study, the source and target corpora are sets of Wikipedia pages related to the source term (S) and its reference translation (T) respectively (see Section 3.1). The degree of corpus preprocessing varies greatly from one study to another. Complex linguistic tools such as terminological extractors (Daille and Morin, 2005), parsers (Yu and Tsujii, 2009) or lemmatizers (Rapp, 1999) are sometimes used.

In our case, the only preprocessing that takes place is the deletion of the Wikipedia symbols pertaining to its particular syntax (*e.g.* `[[ ]]`).[2] It is to be noted that, for the sake of simplicity and generality, our implementation does not exploit interlanguage links nor structural elements specific to Wikipedia documents, as opposed to (Yu and Tsujii, 2009).

Step 2 A context vector v_s for the source term S is built (see Figure 1 for a made-up example). This vector contains the words that are in the context of the occurrences of S and are strongly correlated to S. The definition of "context" is one of the parameters whose best value we want to find. Context length can be based on a number of units, for instance 3 sentences (Daille and Morin, 2005), windows of 3 (Rapp, 1999) or 25 words (Prochasson et al., 2009), etc. It is an important parameter of the projection-based approach. Should the context length be too small, we would miss words that would be relevant in finding the translation. On the other hand, if the context is too large, it

[2]We used a set of about 40 regular expressions to do this.

might contain too much noise. At this step, a stoplist made of function words is applied in order to filter out context words and reduce noise in the context vector.

Additionally, an association measure is used to score the strength of correlation between S and the words in its contexts; it serves to normalize corpus frequencies. Words that have a high association score with S are more prominent in the context vector. The association measure is the second important parameter we want to study. As already noted, most authors use the log-likelihood ratio to measure the association between collocates; some, like (Rapp, 1999), informally compare the performance of a small number of association measures, or combine the results obtained with different association measures (Daille and Morin, 2005).

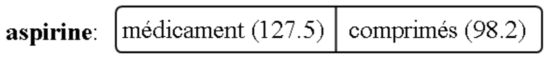


Figure 1: Step 2

Step 3 Words in v_s are projected into the target language with the help of the bilingual seed lexicon (Figure 2). Each word in v_s which is present in the bilingual lexicon is translated, and those translations define the projected context vector v_p. Words that are not found in the bilingual lexicon are simply ignored. The size of the seed lexicon and its content are therefore two important parameters of the approach. In previous studies, seed lexicons vary between 16,000 (Rapp, 1999) and 65,000 (Déjean et al., 2005) entries, a typical size being around 20,000 (Fung, 1998; Chiao and Zweigenbaum, 2002; Daille and Morin, 2005).

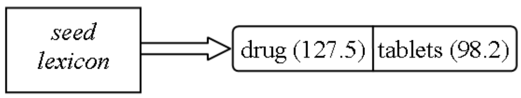


Figure 2: Step 3

Step 4 Context vectors v_t are computed for each candidate term in the target language corpus (Figure 3). The dimension of the target-vector space is defined to be the one induced by the projection mechanism described in Step 3. The context vector v_t of each candidate term is computed as in Step 2. Therefore, in Step 4, the parameters of context definition and association measure are important and take the same values as those in Step 2. Note that in this study, on top of all single terms, we also consider target bigrams as potential candidates (99.5 % of our reference target terms are composed of at most two words). As such, our method can handle complex terms (of up to two words), as opposed to most previous studies, without having to resort to a separate terminological extraction as in (Daille and Morin, 2005).

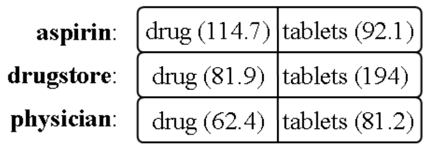


Figure 3: Step 4

Step 5 Context vectors v_t are ranked in decreasing order of their similarity with v_p (Figure 4). The similarity measure between context vectors varies among studies: city-block measure (Rapp, 1999), cosine (Fung, 1998; Chiao and Zweigenbaum, 2002; Daille and Morin, 2005; Prochasson et al., 2009), Dice or Jaccard indexes (Chiao and Zweigenbaum, 2002; Daille and Morin, 2005), etc. It is among the parameters whose effect we experimentally evaluate.

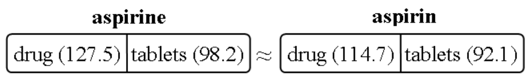


Figure 4: Step 5

2.2 Parameters studied

The five steps we described involve many parameters, the values of which can influence at varying degrees the performance of a translation spotter. In the current study, we considered the following parameter values.

Context We considered contexts defined as the current sentence or the current paragraph involving S. We also considered windows of 5 and 25 words on both sides of S.

Association measure Following the aforementioned studies, we implemented these popular measures: pointwise mutual information (PMI), log-likelihood ratio (LL) and chi-square (χ^2). We also implemented the discounted log-odds (LO) described by (Evert, 2005, p. 86) in his work on collocation mining. To our knowledge, this association measure has not been used yet in translation spotting. It is computed as:

$$\text{odds-ratio}_{disc} = \log \frac{(O_{11} + \frac{1}{2})(O_{22} + \frac{1}{2})}{(O_{12} + \frac{1}{2})(O_{21} + \frac{1}{2})}$$

where O_{ij} are the cells of the 2×2 contingency matrix of a word token s cooccurring with the term S within a given window size.[3]

Similarity measure We implemented four measures: city-block, cosine, as well as Dice and Jaccard indexes (Jurafsky and Martin, 2008, p. 666). Our implementations of Dice and Jaccard are identical to the *DiceMin* and *JaccardMin* similarity measures reported in (Otero, 2008) and which outperformed the other five metrics he tested.

Seed lexicon We investigated the impact of both the size of the lexicon and its content. We started our study with a mixed lexicon of around 5,000 word entries: roughly 2,000 of them belong to the medical domain, while the other entries belong to the general language. We also considered mixed lexicons of 7,000, 9,000 and 11,000 entries (where 2,000 entries are related to the medical domain), as well as a 5,000-entry general language only lexicon.

2.3 Cognate heuristic

Many authors are embedding heuristics in order to improve their approach. For instance, Chiao and Zweigenbaum (2002) propose to integrate a reverse translation spotting strategy in order to improve precision. Prochasson et al. (2009) boost the strength of context words that happen to be transliterated in the other language. A somehow generalized version of this heuristic has been described in (Shao and Ng, 2004).

In this work, we examine the performance of the best configuration of parameters we found, combined with a simple heuristic based on graphic similarity between source and target terms, similar to the orthographic features in (Haghighi et al., 2008)'s generative model. This is very specific to our task where medical terms often (but not always) share Latin or Greek roots, such as *microvillosités* in French and *microvilli* in English.

In this heuristic, translation candidates which are cognates of the source term are ranked first among the list of translation candidates. In our implementation, two words are cognates if their first four characters are identical (Simard et al., 1992). One interesting note concerns the word-order mismatch typically observed in French and English complex terms, such as in *ADN mitochondrial* (French) and *mitochondrial DNA* (English). We do treat this case adequately.

3 Experimental protocol

In order to pinpoint the best configuration of values for the parameters identified in Section 2.2, four series of experiments were carried out. In all of them, the task consists of spotting translation candidates for each source language term using the resources[4] described below. The quality of the results is evaluated with the help of the metrics described in Section 3.2.

3.1 Resources

Corpora The comparable corpora are made of the (at most) 50 French and English Wikipedia documents that are the most relevant to the source term and to its reference translation respectively. These documents are retrieved with the NLGbAse Information Retrieval tool.[5] The average token count of all the 50-document corpora as well as the average frequency of the source and target terms in these corpora for our four series of experiments are listed in Table 1.

[3] For instance, O_{21} stands for the number of windows containing S but not s.

[4] Our resources are available at `http://olst.ling.umontreal.ca/~audrey/coling2010/`. They were acquired as described in (Rubino, 2009).

[5] `http://nlgbase.org/`

	Experiment			
	1	2	3	4
Tokens_s	89,431	73,809	42,762	90,328
Tokens_t	52,002	27,517	12,891	38,929
$\lvert S\rvert$	296	184	66	306
$\lvert T\rvert$	542	255	104	404

Table 1: 50-document corpora averages

The corpora are somewhat small (most corpora in previous studies are made of at least a million words). We believe this is more representative of a task where we try to translate domain specific terms. Some of the Wikipedia documents may contain a handful of parallel sentences (Smith et al., 2010), but this information is not used in our approach. The construction of the corpus involves a bias in that the reference translations are used to obtain the most relevant target language documents. However, since our objective is to compare the relative performance of different sets of parameters, this does not affect our results. In fact, as per (Déjean et al., 2005) (whose comparable corpora are English and German abstracts), the use of such an "ideal" corpus is common (as in (Chiao and Zweigenbaum, 2002), where the corpus is built from a specific query).

Seed lexicon The mixed seed lexicon we use is taken from the Heymans Institute of Pharmacology's *Multilingual glossary of technical and popular medical terms*.[6] Random general language entries from the `FreeLang`[7] project are also incorporated into the lexicon for some of our experiments.

Reference translations The test set is composed of 5,000 nominal single and multi-word pairs of French and English terms from the MeSH (*Medical Subject Heading*) thesaurus.[8]

3.2 Evaluation metrics

The performance of each set of parameters in the experiments is evaluated with Top N precision (P_N), recall (R_N) and F-measure (F_N), as well as Mean Average Precision (MAP). Precision is the number of correct translations (at most 1 per source term) divided by the number of terms for which our system gave at least one answer; recall is equal to the ratio of correct translations to the total number of terms. F-measure is the harmonic mean of precision and recall:

$$\text{F-measure} = \frac{2 \times (precision \times recall)}{(precision + recall)}$$

The MAP represents in a single figure the quality of a system according to various recall levels (Manning et al., 2008, p. 147–148):

$$\text{MAP(Q)} = \frac{1}{|Q|} \sum_{|Q|}^{j=1} \frac{1}{m_j} \sum_{m_j}^{k=1} Precision(R_{jk})$$

where $|Q|$ is the number of terms to be translated, m_j is the number of reference translations for the j^{th} term (always 1 in our case), and $Precision(R_{jk})$ is 0 if the reference translation is not found for the j^{th} term or $1/r$ if it is (r is the rank of the reference translation in the translation candidates).

4 Experiments

In Experiment 1, 500 single and multi-word terms must be translated from French to English using each of the 64 possible configurations of these parameters: context definition, association measure and similarity measure. In Experiment 2, we submit to the 8 best variants 1,500 new terms to determine with greater confidence the best 2, which are again tested on the last 3,000 of the test terms (Experiment 3). In Experiment 4, using 1,350 frequent terms, we examine the effects of seed lexicon size and specificity and we apply a heuristic based on cognates.

4.1 Experiment 1

The results of the first series of experiments on 500 terms can be analysed from the point of view of each of the parameters whose values varied among 64 configurations (Section 2.2). The maximal MAP reached for each parametric value is given in Table 2.

The most notable result is that, of the four association measures studied, the log-odds ratio is

[6] `http://users.ugent.be/~rvdstich/eugloss/welcome.html`

[7] `http://www.freelang.net/`

[8] `http://www.nlm.nih.gov/mesh/`

Param.	Value	Best MAP	In config.
association	LO	**0.536**	sentence_cosine
	LL	0.413	sentence_Dice
	PMI	0.299	sentence_city-block
	χ^2	0.179	sentence_Dice
similarity	cosine	**0.536**	sentence_LO
	Dice	0.520	sentence_LO
	Jaccard	0.520	sentence_LO
	city-block	0.415	sentence_LO
context	sentence	**0.536**	cosine_LO
	paragraph	0.460	cosine_LO
	25 words	0.454	cosine_LO
	5 words	0.361	Dice_LO

Table 2: Best MAP in Experiment 1

significantly superior to the others in every variant. There is as much as 34 % difference between LO and other measures for Top 1 recall. This is interesting since most previous works use the log-likelihood, and none use LO. Our best results for LO (with cosine_sentence) and LL (with Dice_sentence) are in Table 3. Note that the oracle recall is 93 % (7 % of the source and target terms were not in the corpus).

Assoc.	R1	R20	P1	P20	F1	F20	MAP
LO	**39.4**	**84.8**	**42.3**	**91.0**	**40.8**	**87.8**	**0.536**
LL	29.0	75.2	31.3	81.0	30.1	78.0	0.413

Table 3: Best LO and LL configurations scores

Another relevant observation is that the parameters interact with each other. When the similarity measure is cosine, PMI results in higher Top 1 F-scores than LL, but the Top 20 F-scores are better with LL. PMI is better than LL when using city-block as a similarity measure, but LL is better than PMI when using Dice and Jaccard indexes. χ^2 gives off the worst MAP in all but 4 of the 64 parametric configurations.

As for similarity measures, the Dice and Jaccard indexes have identical performances, in accordance with the fact that they are equivalent (Otero, 2008).[9] Influences among parameters are also observable in the performance of similarity measures. When the association measure is LO, the cosine measure gives slightly better Top 1 F-scores, while the Dice index performs slightly better with regards to Top 20 F-scores. Dice is better when the association measure is LL, with a Top 1 F-score gain of about 15 % compared to the cosine.

Again, in the case of context definitions, relative performances depend on the other parameters and on the number of top translation candidates considered. With LO, sentence contexts have the highest Top 1 F-measures, while Top 20 F-measures are highest with paragraphs, and 5-word contexts are the worst.

4.2 Experiment 2

The best parametric values found in Experiment 1 were put to the test on 1,500 different test terms for scale-up verification. Along with LO, which was the best association measure in the previous experiment, we used LL to double-check its relative inefficiency. For all of the 8 configurations evaluated, LL's recall, precision and MAP remain worse than LO's. In particular, LO's MAP scores with the cosine measure are more than twice as high as LL's (respectively 0.33 and 0.124 for sentence contexts). As in Experiment 1, the Dice index is significantly better for LL compared to the cosine, but not for LO. In the case of LO, sentence contexts have better Top 1 performances than paragraphs, and vice versa for Top 20 performances (see Table 4; oracle recall is 93.5 %). Hence, paragraph contexts would be more useful in tasks consisting of proposing candidate translations to lexicographers, while sentences would be more appropriate for automatic bilingual lexicon construction.

Ctx	R1	R20	P1	P20	F1	F20	MAP
Sent.	**23.1**	63.9	**27.8**	76.6	**25.23**	69.68	**0.336**
Parag.	20.1	**70.0**	22.9	**79.7**	21.41	**74.54**	0.325

Table 4: LO_Dice configuration scores

The cosine and Dice similarity measures have similar performances when LO is used. Moreover, we observe the effect of source and target term frequencies in corpus. As seen in Table 1, these frequencies are on average about half smaller in Experiment 2 as they are in Experiment 1, which results in significantly lower performances for all

[9] For this reason, whenever "Dice" is mentioned from this point on, it also applies to the Jaccard index.

8 variants. As Figure 5 shows for the variant LO_cosine_sentence, terms that are more frequent have a greater chance of being correctly translated at better ranks.

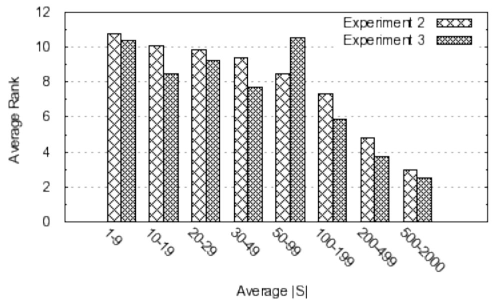


Figure 5: Average rank of correct translation according to average source term frequency

However, the relative performance of the different parametric configurations still holds.

4.3 Experiment 3

In Experiment 3, we evaluate the two best configurations from Experiment 2 with 3,000 new terms in order to verify the relative performance of the cosine and Dice similarity measures. As Table 5 shows, cosine has slightly better Top 1 figures, while Dice is a little better when considering the Top 20 translation candidates. Therefore, as previously mentioned, the choice of similarity measure (cosine or Dice) should depend on the goal of translation spotting. Note that the scores in Experiment 3 are much lower than those of Experiments 1 and 2 because of low term frequencies in the corpus (see Table 1 and Figure 5). Also, oracle recall is only 71.1 %.

Sim.	R1	R20	P1	P20	F1	F20	MAP
Cosine	**9.8**	28.1	**20.7**	59.4	**13.3**	38.15	0.232
Dice	9.4	**28.9**	19.8	**61.2**	12.75	**39.26**	**0.286**

Table 5: LO_sentence configuration scores

4.4 Experiment 4

In the last series of experiments, we examine the influence of the bilingual seed lexicon specificity and size, using the 1,350 terms which have source and target frequencies $\geq$ 30 from the 1,500 and 3,000 sets used in Experiments 2 and 3 (oracle recall: 100 %). We tested the different lexicons (see Section 2.2) on the 4 parametric configurations made of sentence contexts, LO or LL association measures, and cosine or Dice similarity measures.

Yet again, LO is better than LL. MAP scores for LO in all variants are comprised in [0.466–0.489]; LL MAPs vary between 0.135 and 0.146 when the cosine is used and between 0.348 and 0.380 when the Dice index is used.

According to our results, translation spotting is more accurate when the seed lexicon contains (5,000) entries from both the medical domain and general language instead of general language words only, but only by a very small margin. Table 6 shows the results for the configuration LO_cosine_sentence. The fact that the difference

Lex.	R1	R20	P1	P20	F1	F20	MAP
Gen. + med.	**39.3**	87.0	**39.6**	87.6	**39.4**	87.3	**0.473**
Gen. only	38.8	**88.1**	39.0	**88.5**	38.9	**88.3**	0.471

Table 6: LO_cosine_sentence configuration scores

is so small could be explained by our resources' properties. The reference translations from MeSH contain terms that are also used in other domains or in the general language, *e.g.* terms from the category "people" (Névéol and Ozdowska, 2006). Wikipedia documents retrieved by using those references may in turn not belong to the medical domain, in which case medical terms from the seed lexicon are not appropriate. Still, the relatively good performance of the general language-only lexicon supports (Déjean et al., 2005, p. 119)'s claim that general language words are useful when spotting translations of domain specific terms, since the latter can appear in generic contexts.

Lexicon sizes tested are 5,000 (the mixed lexicon used in previous experiments), 7,000, 9,000 and 11,000 entries. The performance (based on MAP) is better when 7,000- and 9,000-entry lexicons are used, because more source language context words can be taken into account. However, when the lexicon reaches 11,000, Top 1 MAP scores and F-measures are slightly lower than those obtained with the 7,000-entry one. This may happen because the lexicon is increased with general language words; 9,000 of the 11,000 entries are not from the medical domain, making it harder for the context words to be specific. It would be interesting to study the specificity of context vectors built from the source corpus. Still, the differences in scores are small, as Table 7 shows (see Table 6 for the results obtained with 5,000 entries). This is because, in our implementation, context vector size is limited to 20, as in (Daille and Morin, 2005), in order to reduce processing time. The influence of context vector sizes should be studied.

Lex. size	R1	R20	P1	P20	F1	F20	MAP
7,000	**41.5**	88.8	**41.6**	89.1	**41.5**	88.9	0.488
9,000	40.9	89.3	41.1	89.7	41.0	89.5	**0.489**
11,000	40.1	**89.8**	40.2	**90.1**	40.1	**89.9**	0.484

Table 7: LO_cosine_sentence configuration scores

The parameters related to the seed lexicon do not have as great an impact on the performance as the choice of association measure does: the biggest difference in F-measures for Experiment 4 is 2.9 %. At this point, linguistic-based heuristics such as graphic similarity should be used to significantly increase performance. We applied the cognate heuristic (Section 2.3) on the Top 20 translation candidates given by the variant LO_sentence_9,000-entry lexicon using cosine and Dice similarity measures. Without the heuristic, Top 1 performances are better with cosine, while Dice is better for Top 20. Applying the cognate heuristic makes the Top 1 precision go from 41.1 % to 55.2 % in the case of cosine, and from 39.6 % to 53.9 % in the case of Dice.

5 Discussion

Our results show that using the log-odds ratio as the association measure allows for significantly better translation spotting than the log-likelihood. A closer look at the translation candidates obtained when using LL, the most popular association measure in projection-based approaches, shows that they are often collocates of the reference translation. Therefore, LL may fare better in an indirect approach, like the one in (Daille and Morin, 2005).

Moreover, we have seen that the cosine similarity measure and sentence contexts give more correct top translation candidates, at least when LO is used. Indeed, the values of the different parameters influence one another in most cases. Parameters related to the seed lexicon (size, domain specificity) are not of great influence on the performance, but this may in part be due to our resources and the way they were built.

The highest Top 1 precision, 55.2 %, was reached with the following parameters: sentence contexts, LO, cosine and a 9,000-entry mixed lexicon, with the use of a cognate heuristic.

In future works, other parameters which influence the performance will be studied, among which the use of a terminological extractor to treat complex terms (Daille and Morin, 2005), more contextual window configurations, and the use of syntactic information in combination with lexical information (Yu and Tsujii, 2009). It would also be interesting to compare the projection-based approaches to (Haghighi et al., 2008)'s generative model for bilingual lexicon acquisition from monolingual corpora.

One latent outcome of this work is that Wikipedia is surprisingly suitable for mining medical terms. We plan to check its adequacy for other domains and verify that LO remains a better association measure for different corpora and domains.

Acknowledgments

We are deeply grateful to Raphaël Rubino who provided us with the data material we have been using in this study. We thank the anonymous reviewers for their suggestions.

References

Chiao, Yun-Chuang and Pierre Zweigenbaum. 2002. Looking for candidate translational equivalents in specialized, comparable corpora. In *19th International Conference on Computational Linguistics*, pages 1208–1212.

Daille, Béatrice and Emmanuel Morin. 2005. French-English terminology extraction from comparable corpora. In *2nd International Joint Conference on Natural Language Processing*, pages 707–718.

Déjean, Hervé, Éric Gaussier, Jean-Michel Renders, and Fatiha Sadat. 2005. Automatic processing of multilingual medical terminology: Applications to thesaurus enrichment and cross-language information retrieval. *Artificial Intelligence in Medicine*, 33(2):111–124. Elsevier Science, New York.

Diab, Mona and Steve Finch. 2000. A statistical word-level translation model for comparable corpora. In *Proceedings of the Conference on Content-Based Multimedia Information Access*.

Dunning, Ted. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74.

Evert, Stefan. 2005. *The Statistics of Word Cooccurrences. Word Pairs and Collocations*. Ph.D. thesis, Universität Stuttgart.

Fung, Pascale. 1995. A pattern matching method for finding noun and proper noun translations from noisy parallel corpora. In *33rd Annual Meeting of the Association for Computational Linguistics*, pages 236–243.

Fung, Pascale. 1998. A statistical view on bilingual lexicon extraction: From parallel corpora to non-parallel corpora. In *3rd Conference of the Association for Machine Translation in the Americas*, pages 1–17.

Haghighi, Aria, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *Human Language Technology and Association for Computational Linguistics*, pages 771–779.

Jurafsky, Daniel and James H. Martin. 2008. *Speech and Language Processing*. Prentice-Hall.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Morin, Emmanuel, Béatrice Daille, Koichi Takeuchi, and Kyo Kageura. 2007. Bilingual terminology mining — using brain, not brawn comparable corpora. In *45th Annual Meeting of the Association for Computational Linguistics*, pages 664–671.

Névéol, Aurélie and Sylwia Ozdowska. 2006. Terminologie médicale bilingue anglais/français: usages cliniques et bilingues. *Glottopol*, 8.

Otero, Pablo Gamallo. 2007. Learning bilingual lexicons from comparable English and Spanish corpora. In *Machine Translation Summit 2007*, pages 191–198.

Otero, Pablo Gamallo. 2008. Evaluating two different methods for the task of extracting bilingual lexicons from comparable corpora. In *1st Workshop Building and Using Comparable Corpora*.

Prochasson, Emmanuel, Emmanuel Morin, and Kyo Kageura. 2009. Anchor points for bilingual lexicon extraction from small comparable corpora. In *Machine Translation Summit XII*, pages 284–291.

Rapp, Reinhard. 1999. Automatic identification of word translations from unrelated English and German corpora. In *37th Annual Meeting of the Association for Computational Linguistics*, pages 66–70.

Rubino, Raphaël. 2009. Exploring context variation and lexicon coverage in projection-based approach for term translation. In *Proceedings of the Student Research Workshop associated with RANLP–09*, pages 66–70.

Saralegi, X., I. San Vicente, and A. Gurrutxaga. 2008. Automatic extraction of bilingual terms from comparable corpora in a popular science domain. In *1st Workshop Building and Using Comparable Corpora*.

Shao, Li and Hwee Tou Ng. 2004. Mining new word translations from comparable corpora. In *20th International Conference on Computational Linguistics*, pages 618–624.

Simard, Michel, George Foster, and Pierre Isabelle. 1992. Using cognates to align sentences in bilingual corpora. In *4th Conference on Theoretical and Methodological Issues in Machine Translation*, pages 67–81.

Smith, Jason R., Chris Quirk, and Kristina Toutanova. 2010. Extracting parallel sentences from comparable corpora using document level alignment. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL*, pages 403–411.

Yu, Kun and Junichi Tsujii. 2009. Bilingual dictionary extraction from Wikipedia. In *Machine Translation Summit XII*.

Constituent Reordering and Syntax Models for English-to-Japanese Statistical Machine Translation

Young-Suk Lee
IBM Research
ysuklee@us.ibm.com

Bing Zhao
IBM Research
zhaob@us.ibm.com

Xiaoqiang Luo
IBM Research
xiaoluo@us.ibm.com

Abstract

We present a constituent parsing-based reordering technique that improves the performance of the state-of-the-art English-to-Japanese phrase translation system that includes distortion models by 4.76 BLEU points. The phrase translation model with reordering applied at the pre-processing stage outperforms a syntax-based translation system that incorporates a phrase translation model, a hierarchical phrase-based translation model and a tree-to-string grammar. We also show that combining constituent reordering and the syntax model improves the translation quality by additional 0.84 BLEU points.

1 Introduction

Since the seminal work by (Wu, 1997) and (Yamada and Knight, 2001), there have been great advances in syntax-based statistical machine translation to accurately model the word order distortion between the source and the target languages.

Compared with the IBM source-channel models (Brown et al., 1994) and the phrase translation models (Koehn et al., 2003), (Och and Ney, 2004) which are good at capturing local reordering within empirical phrases, syntax-based models have been effective in capturing the long-range reordering between language pairs with very different word orders like Japanese-English (Yamada and Knight, 2001), Chinese-English (Chiang, 2005) and Urdu-English (Zollmann et al. 2008), (Callison-Burch et al. 2010).

However, (Xu et al., 2009) show that applying dependency parsing-based reordering as pre-processing (pre-ordering hereafter) to phrase translation models produces translation qualities significantly better than a hierarchical phrase-based translation model (Hiero hereafter) implemented in (Zollman and Venugopal, 2006) for English-to-Japanese translation. They also report that the two models result in comparable translation qualities for English-to-Korean/Hindi/Turkish/Urdu, underpinning the limitations of syntax-based models for handling long-range reordering exhibited by the strictly head-final Subject-Object-Verb (SOV) order languages like Japanese and the largely head-initial Subject-Verb-Object (SVO) order languages like English.

In this paper, we present a novel constituent parsing-based reordering technique that uses manually written context free (CFG hereafter) and context sensitive grammar (CSG hereafter) rules. The technique improves the performance of the state-of-the-art English-to-Japanese phrase translation system that includes distortion models by 4.76 BLEU points. The phrase translation model with constituent pre-ordering consistently outperforms a syntax-based translation system that integrates features from a phrase translation model, Hiero and a tree-to-string grammar. We also achieve an additional 0.84 BLEU point improvement by applying an extended set of reordering rules that incorporate new rules learned from the syntax model for decoding.

The rest of the paper is organized as follows. In Section 2, we summarize previous work related to this paper. In Section 3, we give an overview of the syntax model with which we compare the performance of a phrase translation

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 626–634,
Beijing, August 2010

model with pre-ordering. We also discuss a chart-based decoder used in all of our experiments. In Section 4, we describe the constituent parsing-based reordering rules. We show the impact of pre-ordering on a phrase translation model and compare its performance with the syntax model. In Section 5, we discuss experimental results from the combination of syntax model and pre-ordering. Finally in Section 6, we discuss future work.

2 Related Work

Along the traditions of unsupervised learning by (Wu, 1997), (Chiang, 2005) presents a model that uses hierarchical phrases, Hiero. The model is a synchronous context free grammar learned from a parallel corpus without any linguistic annotations and is applied to Chinese-to-English translation. (Galley and Manning, 2008) propose a hierarchical phrase reordering model that uses shift-reduce parsing.

In line with the syntax-based model of (Yamada and Knight, 2001) that transforms a source language parse tree into a target language string for Japanese-English translation, linguistically motivated syntactic features have been directly incorporated into both modeling and decoding. (Liu, et. al. 2006), (Zhao and Al-Onaizan, 2008) propose a source tree to target string grammar (tree-to-string grammar hereafter) in order to utilize the source language parsing information for translation. (Liu, et. al. 2007) propose packed forest to allow ambiguities in the source structure for the tree-to-string grammar. (Ding and Palmer, 2005) and (Zhang et. al., 2006) propose a tree-to-tree grammar, which generates the target tree structure from the high-precision source syntax. (Shen, et. al., 2008) propose a string to dependency tree grammar to use the target syntax when the target is English for which parsing is more accurate than other languages. (Marcu et al., 2006) introduce a syntax model that uses syntactified target language phrases. (Chang and Toutanova, 2007) propose a global discriminative statistical word order model that combines syntactic and surface movement information, which improves the translation quality by 2.4 BLEU points in English-to-Japanese translation. (Zollmann, et. al., 2008) compare various translation models and report that the syntax augmented model works better for Chinese-to-English and Urdu-to-English, but not for Arabic-to-English translation. (Carreras and Collins, 2009) propose a highly flexible reordering operations during tree adjoining grammar parsing for German-English translation. (Callison-Burch et al., 2010) report a dramatic impact of syntactic translation models on Urdu-to-English translation.

Besides the approaches which integrate the syntactic features into translation models, there are approaches showing improvements via pre-ordering for model training and decoding. (Xia and McCord, 2004), (Collins et al., 2005) and (Wang, et. al. 2007) apply pre-ordering to the training data according to language-pair specific reordering rules to improve the translation qualities of French-English, German-English and Chinese-English, respectively. (Habash, 2007) uses syntactic preprocessing for Arabic-to-English translation. (Xu et al., 2009) use a dependency parsing-based pre-ordering to improve translation qualities of English to five SOV languages including Japanese.

The current work is related to (Xu et al., 2009) in terms of the language pair and translation models explored. However, we use constituent parsing with hierarchical rules, while (Xu et al., 2009) use dependency parsing with precedence rules. The two approaches have different rule coverage and result in different word orders especially for phrases headed by verbs and prepositions. We also present techniques for combining the syntax model with tree-to-string grammar and pre-ordering for additional performance improvement. The total improvement by the current techniques over the state-of-the-art phrase translation model is 5.6 BLEU points, which is an improvement gap not attested elsewhere with reordering approaches.

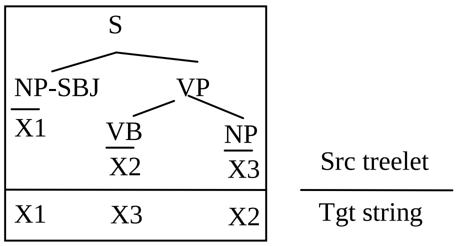
3 Syntax Model and Chart-Based Decoder

In this section, we give an overview of the syntax model incorporating a tree-to-string grammar. We will compare the syntax model performance with a phrase translation model that uses the pre-ordering technique we propose in Section 4. We also describe the chart-based decoder that we use in all of the experiments reported in this paper.

3.1 Tree-to-String Grammar

Tree-to-string grammar utilizes the source language parse to model reordering probabilities from a source tree to the target string (Liu et. al., 2006), (Liu et. al., 2007), (Zhao and Al-Onaizan, 2008) so that long distance word reordering becomes local in the parse tree.

Reordering patterns of the source language syntax and their probabilities are automatically learned from the word-aligned source-parsed parallel data and incorporated as a tree-to-string grammar for decoding. Source side parsing and the resulting reordering patterns bound the search space. Parsing also assigns linguistic labels to the chunk, e.g. NP-SBJ, and allows statistics to be clustered reasonably. Each synchronous context free grammar (SCFG) rewriting rule rewrites a source treelet into a target string, with both sides containing hiero-style variables. For instance, the rule [X, VP] [X, VB] [X,NP] → [X, NP] [X, VB] rewrites a VP with two constituents VB and NP into an NP VB order in the target, shown below.



The tree-to-string grammar introduces possible search space to generate an accurate word order, which is refined on the basis of supports from other models. However, if the correct word order cannot be generated by the tree-to-string grammar, the system can resort to rules from Hiero or a phrase translation model for extended rule coverage.

3.2 Chart-based Decoder

We use a chart-based decoder − a template decoder that generalizes over various decoding schemes in terms of the dot-product in Earley-style parsing (Earley, 1970) − to support various decoding schemes such as phrase, Hiero (Chiang, 2005), Tree-to-String, and the mixture of all of the above.

This framework allows one to strictly compare different decoding schemes using the same feature and parameter setups. For the experimental results in Sections 4 & 5, we applied (1) phrase decoding for the baseline phrase translation system that includes distortion models, (2) Hiero decoding for the Hiero system that incorporates a phrase translation model, and (3) Tree-to-string decoding for the syntax-based systems that incorporate features from phrase translation, Hiero and tree-to-string grammar models.

The decoder seeks the best hypothesis e^* according to the Bayesian decision rule (1):

$$e^* = \operatorname*{argmin}_{\{e,d\in D\}} \phi(e)\cdot\phi(d) \qquad (1)$$

d is one derivation path, rewriting the source tree into the target string via the probabilistic synchronous context free tree-to-string grammar (PSCFG). $\phi(e)$ is the cost functions computed from general n-gram language models. In this work, we use two sets of interpolated 5-gram language models. $\phi(d)$ is a vector of cost functions defined on the derivation sequence. We have integrated 18 cost functions ranging from the basic relative frequencies and IBM model-1 scores to counters for different types of rules including blocks, glue, Hiero, and tree-to-string grammar rules. Additional cost functions are also integrated into the decoder, including measuring the function/content-word mismatch between source and target, similar to (Chiang et. al., 2009) and length distribution for non-terminals in (Shen et. al., 2009).

4 Parsing and Reordering Rules

We apply a set of manually acquired reordering rules to the parsing output from a constituent parser to pre-order the data for model training and decoding.

4.1 Parsing with Functional Tags

We use a maximum entropy English parser (Ratnaparkhi, 1999) trained on OntoNotes (Hovy, 2006) data. OntoNotes data include most of the Wall Street Journal data in Penn Treebank (Marcus et al., 1993) and additional data from broadcast conversation, broadcast news and web log.

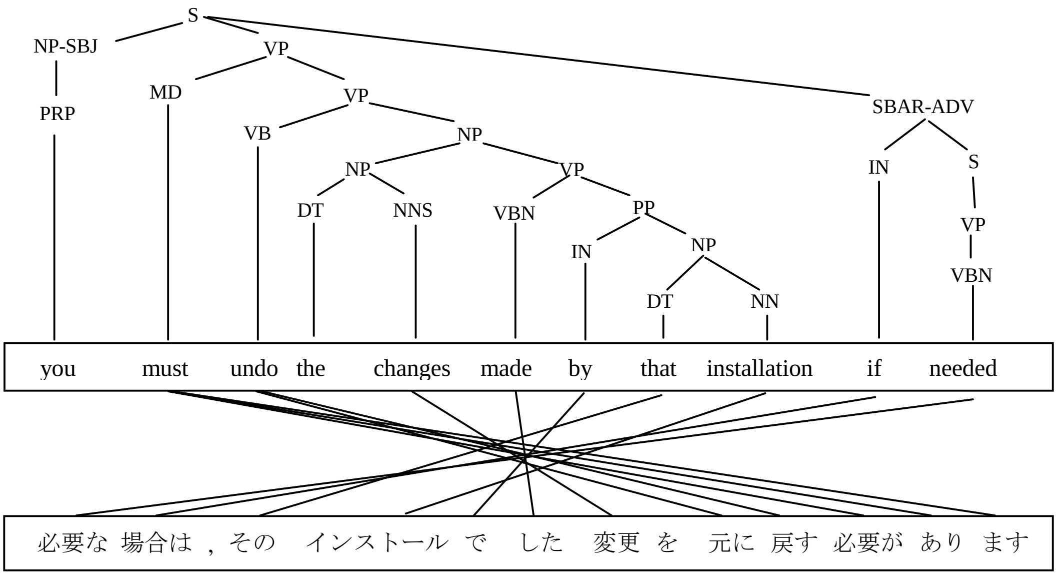


Figure 1. Parse Tree and Word Alignment before Reordering

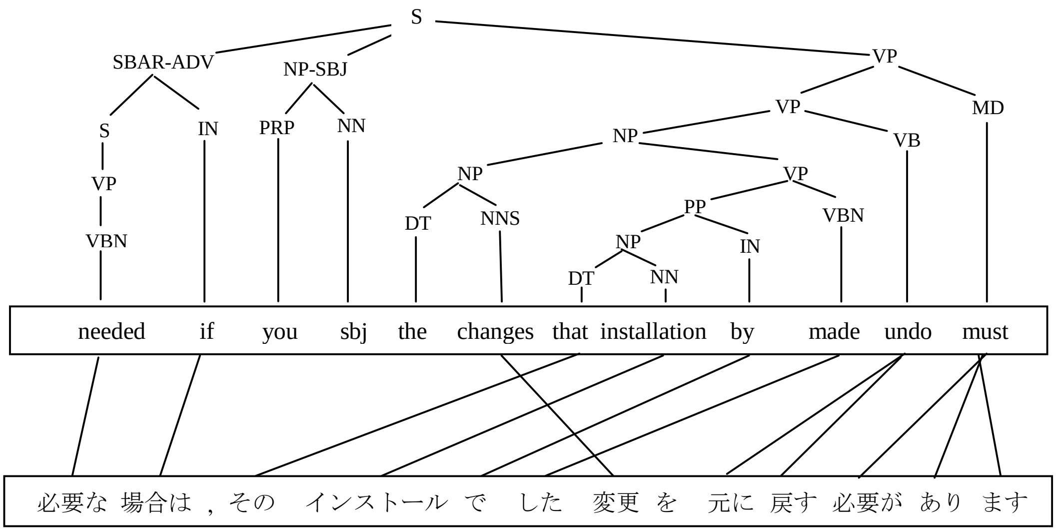


Figure 2. Parse Tree and Word Alignment after Reordering

The parser is trained with all of the functional and part-of-speech (POS) tags in the original distribution: total 59 POS tags and 364 phrase labels.

We use functional tags since reordering decisions for machine translation are highly influenced by the function of a phrase, as will be shown later in this section. An example parse tree with functional tags is given at the top half of Figure 1. NP-SBJ indicates a subject noun phrase, SBAR-ADV, an adverbial clause.

4.2 Structural Divergence between English and Japanese

Japanese is a strictly head-final language, i.e. the head is located at the end of a phrase. This leads to a high degree of distortions with English, which is largely head initial.

The word order contrast between the two languages is illustrated by the human word alignment at the bottom half of Figure 1. All instances of word alignments are non-monotonic except for the sequence *that installation*, which is monotonically aligned to the Japanese morpheme sequence その インストール. Note that there are no word boundaries in Japanese written text, and we apply Japanese morpheme segmentation to obtain morpheme sequences in the figure. All of the Japanese examples in this paper are presented with morpheme segmentation.

The manual reordering rules are written by a person who is proficient with English and Japanese/Korean grammars, mostly on the basis of perusing parsed English texts.

4.3 CFG Reordering Rules

Our reordering rules are mostly CFG rules and divided into head and modifier rules.

Head reordering rules in Table 1 move verbs and prepositions from the phrase initial to the phrase final positions (Rules 1-11). Reordering of the head phrase in an adverbial clause also belongs to this group (Rules 12-14). The label sequences in **Before RO** and **After RO** are the immediate children of the **Parent Node** before and after reordering. VBX stands for VB, VBZ, VBP, VBD, VBN and VBG. XP^+ stands for one or more POS and/or phrase labels such as MD, VBX, NP, PP, VP, etc. In 2 & 4, RB is the tag for negation *not*. In 5, RP is the tag for a verb particle.

Modifier reordering rules in Table 2 move modified phrases from the phrase initial to the phrase final positions within an NP (Rules 1-3). They also include placement of NP, PP, ADVP within a VP (Rules 4 & 5). The subscripts in a rule, e.g. PP_1 and PP_2 in Rule 3, indicate the distinctness of each phrase sharing the same label.

4.4 CSG Reordering Rules

Some reordering rules cannot be captured by CFG rules, and we resort to CSG rules.[1]

	Parent Node	**Before RO**	**After RO**
1	VP	MD VP	VP MD
2	VP	MD RB VP	VP MD RB
3	VP	VBX XP^+	XP^+ VBX
4	VP	VBX RB XP^+	XP^+ VBX RB
5	VP	VBX RP XP^+	XP^+ VBX RP
6	ADJP-PRD	JJ XP^+	XP^+ JJ
7	PP	IN NP	NP IN
8	PP	IN S	S IN
9	SBAR-TMP	IN S	S IN
10	SBAR-ADV	IN S	S IN
11	SBAR-PRP	IN S	S IN
12	SBAR-TMP	WHADVP S	S WHADVP
13	SBAR-ADV	WHADVP S	S WHADVP
14	SBAR-PRP	WHADVP S	S WHADVP

Table 1. Head Reordering Rules

	Parent Node	**Before RO**	**After RO**
1	NP	NP SBAR	SBAR NP
2	NP	NP PP	PP NP
3	NP	NP PP_1 PP_2	PP_1 PP_2 NP
4	VP	VBX NP PP	PP NP VBX
5	VP	VBX NP ADVP-TMP PP	PP NP ADVP-TMP VBX

Table 2. Modifier Reordering Rules

For instance, in the parse tree and word alignment in Figure 1, the last two English words *if needed* under SBAR-ADV is aligned to the first two Japanese words 必要な 場合は. In order to change the English order to the corresponding Japanese order, SBAR-ADV dominated by the VP should move across the VP to sentence initial position, as shown in the top half of Figure 2, requiring a CSG rule.

The adverbial clause reordering in Figure 2 is denoted as Rule 1 in Table 3, which lists two other CSG rules, Rule 2 & 3.[2] The subscripts in Table 3 are interpreted in the same way as those in Table 2.

[1] These CSG rules apply to trees of depth two or more, and the applications are dependent on surrounding contexts. Therefore, they are different from CFG rules which apply only to trees of depth one, and TSG (tree substitution grammar) rules for which variables are independently substituted by substitution. The readers are referred to (Joshi and Schabes, 1997) for formal definitions of various grammar formalisms.

[2] Rule 3 is applied after all CFG rules, see Section 4.6. Therefore, VBX's are located at the end of each corresponding VP.

	Before RO → After RO
1	(S XP_1^+ (VP XP_2^+ SBAR-ADV)) → (S SBAR-ADV XP_1^+ (VP XP_2^+))
2	(S XP_1^+ (VP (XP_2^+ SBAR-ADV))) → (S XP_1^+ SBAR-ADV (VP (XP_2^+)))
3	(VP_1 ADVP-MNR (VP_2 XP^+ VBX_2) VBX_1) → (VP_1 (VP_2 XP^+ ADVP-MNR VBX_2) VBX_1)

Table 3. CSG Reordering Rules

ADVP-MNR stands for a manner adverbial phrase such as *explicitly* in the following: *The software version has been explicitly verified as working*. Rule 3 in Table 3 indicates that a ADVP-MNR has to immediately precede a verb in Japanese, resulting in the substring '*...as working explicitly verified...*' after reordering.

Note that functional tags allow us to write reordering rules specific to semantic phrases. For instance, in Rule 1, SBAR-ADV under VP moves to the sentence initial position under S, but an SBAR without any functional tags do not. It typically stays within a VP as the complement of the verb.

4.5 Subject Marker Insertion

Japanese extensively uses case particles that denote the role of the preceding noun phrase, for example, as subject, object, etc. We insert *sbj*, denoting the subject marker, at the end of a subject noun phrase NP-SBJ. The inserted subject marker *sbj* mostly gets translated into the subject particle が or は in Japanese.[3]

4.6 Reordering Rule Application

The rules are applied categorically, sequentially and recursively. CSG Rules 1 and 2 in Table 3 are applied before all of the CFG rules. Among CFG rules, the modifier rules in Table 2 are applied before the head rules in Table 1. CSG Rule 3 in Table 3 is applied last, followed by the subject marker insertion operation.

CFG head and modifier rules are applied recursively. The top half of Figure 2 is the parse tree obtained by applying reordering rules to the parse tree in Figure 1. After reordering, the word alignment becomes mostly monotonic, as shown at the bottom half of Figure 2.

[3] The subject marker insertion is analogous to the insertion operation in (Yamada and Knight, 2001), which covers a wide range of insertion of case particles and verb inflections in general.

4.7 Experimental Results

All systems are trained on a parallel corpus, primarily from the Information Technology (IT) domain and evaluated on the data from the same domain. The training data statistics is in Table 4 and the evaluation data statistics is in Table 5. Japanese tokens are morphemes and English tokens are punctuation tokenized words.

Corpus Stats	English	Japanese
sentence count	3,358,635	3,358,635
token count	57,231,649	68,725,865
vocabulary size	242,712	348,221

Table 4. Training Corpus Statistics

Data Sets	Sentence Count	Token Count
Tuning	600	11,761
DevTest	437	8,158
Eval	600	11,463

Table 5. Evaluation Data Statistics

We measure the translation quality with IBM BLEU (Papineni et al., 2002) up to 4 grams, using 2 reference translations, BLEUr2n4. For BLEU score computation, we character-segment Kanji and Kana sequences in the reference and the machine translation output. Various system performances are shown in Table 6.

Models	Tuning	DevTest	Eval
Phrase (BL)	0.5102	0.5330	0.5486
Hiero	0.5385	0.5574	0.5724
Syntax	0.5561	0.5777	0.5863
Phrase+RO1	0.5681	0.5793	0.5962

Table 6. Model Performances (BLEUr2n4)

Phrase (BL) is the baseline phrase translation system that incorporates lexical distortion models (Al-Onaizan and Papineni, 2006). Hiero is the hierarchical phrase-based system (Chiang, 2006) that incorporates the phrase translation model. Syntax is the syntax model described in Section 3, which incorporates the phrase translation, Hiero and tree-to-string grammar models. Phrase+RO1 is the phrase translation model with pre-ordering for system training and decoding, using the rules described in this section. Phrase+RO1 improves the translation quality of the baseline model by 4.76 BLEU points and outperforms the syntax model by over 0.9 BLEU points.

5 Constituent Reordering and Syntax Model Combined

Translation qualities of systems that combine the syntax model and pre-ordering are shown in Table 7. Syntax+RO1 indicates the syntax model with pre-ordering discussed in Section 4. Syntax+RO2 indicates the syntax model with a more extensive pre-ordering for decoding discussed below .

Models	Tuning	DevTest	Eval
Phrase+RO1	0.5681	0.5793	0.5962
Syntax+RO1	0.5742	0.5802	0.6003
Syntax+RO2	0.5769	0.5880	0.6046

Table 7. Syntax Model with Pre-ordering

Analyses of the syntax model in Table 6 revealed that automatically learned rules by the tree-to-string grammar include new rules not covered by the manually written rules, some of which are shown in Table 8.

Parent Node	**Before RO**	**After RO**
ADJP-PRD	RB JJ PP	PP RB JJ
ADVP-TMP	RB PP	PP RB
ADVP	ADVP PP	PP ADVP
NP	NP VP	VP NP

Table 8. New CFG rules automatically learned by Tree-to-String grammar

We augment the manual rules with the new automatically learned rules. We call this extended set of reordering rules RO2. We use the manual reordering rules RO1 for model training, but use the extended rules RO2 for decoding. And we obtain the translation output Syntax+RO2 in Table 7. Syntax+RO2 outperforms Phrase+RO1 by 0.84 BLEU points, and Syntax+RO1 by 0.43 BLEU points.

In Table 9, we show the ratio of each rule type preserved in the derivation of one-best translation output of the following two models: Syntax and Syntax+RO2. In the table, 'Blocks' indicate phrases from the phrase translation model and 'Glue Rules' denote the default grammar rule for monotone decoding.

The syntax model without pre-ordering (Syntax) heavily utilizes the Hiero and tree-to-string grammar rules, whereas the syntax model with pre-ordering (Syntax+RO2) mostly depends on monotone decoding with blocks and glue rules.

Rule Type	Syntax	Syntax+RO2
Blocks	46.3%	51.2%
Glue Rules	6.0%	37.3%
Hiero Rules	18.3%	1.3%
Tree-to-String	29.4%	10.2%

Table 9. Ratio of each rule type preserved in the translation derivation of Syntax and Syntax+RO2

6 Summary and Future Research

We have proposed a constituent pre-ordering technique for English-to-Japanese translation. The technique improves the performance of the state-of-the-art phrase translation models by 4.76 BLEU points and outperforms a syntax-based translation system that incorporates a phrase translation model, Hiero and a tree-to-string grammar. We have also shown that combining constituent pre-ordering and the syntax model improves the translation quality by additional 0.84 BLEU points.

While achieving solid performance improvement over the existing translation models for English-to-Japanese translation, our work has revealed some limitations of syntax models both in terms of grammar representations and modeling. Whereas many syntax models are based on CFG rules for probability acquisition, the current research shows that there are various types of reordering that require the generative capacity beyond CFG. While most of the reordering rules for changing the English order to the Japanese order (and vice versa) should apply categorically,[4] often the probabilities of tree-to-string grammar rules are not high enough to survive in the translation derivations.

As for the reordering rules that require the generative capacity beyond CFG, we may model mildly context sensitive grammars such as tree adjoining grammars (Joshi and Schabes, 1997), as in (Carreras and Collins, 2009). The

[4] Assuming that the parses are correct, the head reordering rules in Table 1 have to apply categorically to change the English order into the Japanese order because English is head initial and Japanese is head final without any exceptions. Similarly, most of the modifier reordering rules in Table 2 have to apply categorically because most modifiers follow the modified head phrase in English, e.g. a relative clause modifier follows the head noun phrase, a prepositional phrase modifier follows the head noun phrase, etc., whereas modifier phrases precede the modified head phrases in Japanese.

extended domain of locality of tree adjoining grammars should suffice to capture non-CFG reordering rules for many language pairs. Alternatively, we can adopt enriched feature representations so that a tree of depth one can actually convey information on a tree of several depths, such as parent annotation of (Klein and Manning, 2003).

Regarding the issue of modeling, we can introduce a rich set of features, as in (Ittycheriah and Roukos, 2007), the weights of which are trained to ensure that the tree-to-string grammar rules generating the accurate target orders are assigned probabilities high enough not to get pruned out in the translation derivation.

Acknowledgement

We would like to acknowledge IBM RTTS (Realtime Translation Systems) team for technical discussions on the topic and the provision of linguistic resources. We also would like to thank IBM SMT (Statistical Machine Translation) team for various software tools and the anonymous reviewers for their helpful comments .

References

Y. Al-Onaizan and K. Papineni. 2006. Distortion models for statistical machine translation. *Proceedings of ACL-COLING*. Pages 529-536.

C. Baker, S. Bethard, M. Bloodgood, R. Brown, C. Callison-Burch, G. Coppersmith, B. Dorr, W. Filardo, K. Giles, A. Irvine, M. Kayser, L. Levin, J. Martineau, J. Mayfield, S. Miller, A. Phillips, A. Philpot, C. Piatko, L. Schwartz, D. Zajic. 2010. Semantically Informed Machine Translation (SIMT). Final Report of the 2009 Summer Camp for Applied Language Exploration.

P. Brown, V. Della Pietra, S. Della Pietra, and R. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation, *Computational Linguistics*, 19(2):263−311.

X. Carreras and M. Collins. 2009. Non-projective parsing for statistical machine translation. *Proceedings of the 2009 EMNLP*. Pages 200-209.

P. Chang and C. Toutanova. 2007. A Discriminative Syntactic Word Order Model for Machine Translation. *Proceedings of ACL*. Pages 9-16.

D. Chiang. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation. *Proceedings of ACL*. Pages 263-270.

D. Chiang, W. Wang and Kevin Knight. 2009. 11,001 new features for statistical machine translation. *Proceedings of HLT-NAACL*. Pages 218-226.

M. Collins, P. Koehn, I. Kucerova. 2005. Clause Restructuring for Statistical Machine Translation. *Proceedings of ACL*. Pages 531-540.

Y. Ding and M. Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. *Proceedings of ACL*. Pages 541-548.

J. Earley. 1970. An efficient context-free parsing algorithm. *Communications of the ACM*. Vol. 13. Pages 94–102.

M. Galley and C. Manning. 2008. A Simple and Effective Hierarchical Phrase Reordering Model. *Proceedings of EMNLP*.

N. Habash. 2007. Syntactic Preprocessing for Statistical Machine Translation. *Proceedings of the Machine Translation Summit*.

E. Hovy, M. Marcus, M. Palmer, L. Ramshaw and R. Weischedel. 2006. OntoNotes: The 90% Solution. *Proceedings of HLT*. Pages 57-60.

A. Ittycheriah and S. Roukos. 2007. Direct Translation Model 2. *Proceedings of HLT-NAACL*.

A. Joshi and Y. Schabes. 1997. Tree-adjoining grammars. In G. Rozenberg and K. Salomaa, editors, *Handbook of Formal Languages*, volume 3. Springer.

D. Klein and C. Manning. 2003. Accurate Unlexicalized Parsing. *Proceedings of 41st ACL*.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation, *Proceedings of HLT−NAACL*. Pages 48−54.

Y. Liu, Q. Liu and S. Lin. 2006. Tree-to-string alignment template for statistical machine translation. *Proceedings of ACL-COLING*.

Y. Liu, Y. Huang, Q. Liu, and S. Lin. 2007. Forest-to-string statistical translation rules. *Proceedings of the 45th ACL*.

D. Marcu, W. Wang, A. Echihabi and K. Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. *Proceedings of EMNLP*. Pages 44-52.

M. Marcus, B. Santorini and M. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2): 313-330.

F. J. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*: Vol. 30. Pages 417– 449.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. *Proceddings of ACL*. Pages 311–318.

A. Ratnaparkhi. 1999. Learning to Parse Natural Language with Maximum Entropy Models. *Machine Learning: Vol. 34*. Pages 151-178.

L. Shen, J. Xu and R. Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. *Proceedings of ACL*.

L. Shen, J. Xu, B. Zhang, S. Matsoukas and Ralph Weischedel. 2009. Effective Use of Linguistic and Contextual Information for Statistical Machine Translation. *Proceedings of EMNLP*.

C. Wang, M. Collins, P. Koehn. 2007. Chinese Syntactic Reordering for Statistical Machine Translation. *Proceedings of EMNLP-CoNLL*.

D. Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3): 377-404.

F. Xia and M. McCord. 2004. Improving a Statistical MT System with Automatically Learned Rewrite Patterns. *Proceedings of COLING*.

P. Xu, J. Kang, M. Ringgaard, F. Och. 2009. Using a dependency parser to improve SMT for subject-verb-object languages. *Proceedings of HLT-NAACL*.

K. Yamada and K. Knight. 2001. A Syntax-based Statistical Translation Model. *Proceedings of the 39th ACL*. Pages 523-530.

H. Zhang, L. Huang, D. Gildea and K. Knight. 2006. Synchronous binarization for machine translation. *Proceedings of the HLT-NAACL*. Pages 256-263.

B. Zhao and Y. Al-onaizan. 2008. Generalizing Local and Non-Local Word-Reordering Patterns for Syntax-Based Machine Translation. *Proceedings of EMNLP*. Pages 572-581.

A. Zollmann and A. Venugopal. 2006. Syntax augmented machine translation via chart parsing. *Proceedings of NAACL 2006 -Workshop on statistical machine translation*.

A. Zollmann, A. Venugopal, F. Och and J. Ponte. 2008. A Systematic Comparison of Phrase-Based, Hierarchical and Syntax-Augmented MT. *Proceedings of COLING*.

Sentiment Classification and Polarity Shifting

Shoushan Li[†‡] **Sophia Yat Mei Lee**[†] **Ying Chen**[†] **Chu-Ren Huang**[†] **Guodong Zhou**[‡]

[†]Department of CBS
The Hong Kong Polytechnic University
{shoushan.li, sophiaym, chenying3176, churenhuang}@gmail.com

[‡] Natural Language Processing Lab
School of Computer Science and Technology
Soochow University
gdzhou@suda.edu.cn

Abstract

Polarity shifting marked by various linguistic structures has been a challenge to automatic sentiment classification. In this paper, we propose a machine learning approach to incorporate polarity shifting information into a document-level sentiment classification system. First, a feature selection method is adopted to automatically generate the training data for a binary classifier on polarity shifting detection of sentences. Then, by using the obtained binary classifier, each document in the original polarity classification training data is split into two partitions, polarity-shifted and polarity-unshifted, which are used to train two base classifiers respectively for further classifier combination. The experimental results across four different domains demonstrate the effectiveness of our approach.

1 Introduction

Sentiment classification is a special task of text classification whose objective is to classify a text according to the sentimental polarities of opinions it contains (Pang et al., 2002), e.g., *favorable* or *unfavorable*, *positive* or *negative*. This task has received considerable interests in the computational linguistic community due to its potential applications.

In the literature, machine learning approaches have dominated the research in sentiment classification and achieved the state-of-the-art performance (e.g., Kennedy and Inkpen, 2006; Pang et al., 2002). In a typical machine learning approach, a document (text) is modeled as a bag-of-words, i.e. a set of content words without any word order or syntactic relation information. In other words, the underlying assumption is that the sentimental orientation of the whole text depends on the sum of the sentimental polarities of content words. Although this assumption is reasonable and has led to initial success, it is linguistically unsound since many function words and constructions can shift the sentimental polarities of a text. For example, in the sentence '*The chair is not comfortable*', the polarity of the word '*comfortable*' is positive while the polarity of the whole sentence is reversed because of the negation word '*not*'. Therefore, the overall sentiment of a document is not necessarily the sum of the content parts (Turney, 2002). This phenomenon is one main reason why machine learning approaches fail under some circumstances.

As a typical case of polarity shifting, negation has been paid close attention and widely studied in the literature (Na et al., 2004; Wilson et al., 2009; Kennedy and Inkpen, 2006). Generally, there are two steps to incorporate negation information into a system: negation detection and negation classification. For negation detection, some negation trigger words, such as '*no*', '*not*', and '*never*', are usually applied to recognize negation phrases or sentences. As for negation classification, one way to import negation information is to directly reverse the polarity of the words which contain negation trigger words as far as term-counting approaches are considered (Kennedy and Inkpen, 2006). An alternative way is to add some negation features (e.g., negation bigrams or negation phrases) into

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 635–643,
Beijing, August 2010

machine learning approaches (Na et al., 2004). Such approaches have achieved certain success.

There are, however, some shortcomings with current approaches in incorporating negation information. In terms of negation detection, firstly, the negation trigger word dictionary is either manually constructed or relies on existing resources. This leads to certain limitations concerning the quality and coverage of the dictionary. Secondly, it is difficult to adapt negation detection to other languages due to its language dependence nature of negation constructions and words. Thirdly, apart from negation, many other phenomena, e.g., contrast transition with trigger words like '*but*', '*however*', and '*nevertheless*', can shift the sentimental polarity of a phrase or sentence. Therefore, considering negation alone is inadequate to deal with the polarity shifting problem, especially for document-level sentiment classification.

In terms of negation classification, although it is easy for term-counting approaches to integrate negation information, they rarely outperform a machine learning baseline (Kennedy and Inkpen, 2006). Even for machine learning approaches, although negation information is sometimes effective for local cases (e.g., *not good*), it fails on long-distance cases (e.g., *I don't think it is good*).

In this paper, we first propose a feature selection method to automatically generate a large scale polarity shifting training data for polarity shifting detection of sentences. Then, a classifier combination method is presented for incorporating polarity shifting information. Compared with previous ones, our approach highlights the following advantages：First of all, we apply a binary classifier to detect polarity shifting rather than merely relying on trigger words or phrases. This enables our approach to handle different kinds of polarity shifting phenomena. More importantly, a feature selection method is presented to automatically generate the labeled training data for polarity shifting detection of sentences.

The remainder of this paper is organized as follows. Section 2 introduces the related work of sentiment classification. Section 3 presents our approach in details. Experimental results are presented and analyzed in Section 4. Finally, Section 5 draws the conclusion and outlines the future work.

2 Related Work

Generally, sentiment classification can be performed at four different levels: word level (Wiebe, 2000), phrase level (Wilson et al., 2009), sentence level (Kim and Hovy, 2004; Liu et al., 2005), and document level (Turney, 2002; Pang et al., 2002; Pang and Lee, 2004; Riloff et al., 2006). This paper focuses on document-level sentiment classification.

In the literature, there are mainly two kinds of approaches on document-level sentiment classification: term-counting approaches (lexicon-based) and machine learning approaches (corpus-based). Term-counting approaches usually involve deriving a sentiment measure by calculating the total number of negative and positive terms (Turney, 2002; Kim and Hovy, 2004; Kennedy and Inkpen, 2006). Machine learning approaches recast the sentiment classification problem as a statistical classification task (Pang and Lee, 2004). Compared to term-counting approaches, machine learning approaches usually achieve much better performance (Pang et al., 2002; Kennedy and Inkpen, 2006), and have been adopted to more complicated scenarios, such as domain adaptation (Blitzer et al., 2007), multi-domain learning (Li and Zong, 2008) and semi-supervised learning (Wan, 2009; Dasgupta and Ng, 2009) for sentiment classification.

Polarity shifting plays a crucial role in phrase-level, sentence-level, and document-level sentiment classification. However, most of previous studies merely focus on negation shifting (polarity shifting caused by the negation structure). As one pioneer research on sentiment classification, Pang et al. (2002) propose a machine learning approach to tackle negation shifting by adding the tag 'not' to every word between a negation trigger word/phrase (e.g., *not*, *isn't*, *didn't*, etc.) and the first punctuation mark following the negation trigger word/phrase. To their disappointment, considering negation shifting has a negligible effect and even slightly harms the overall performance. Kennedy and Inkpen (2006) explore negation shifting by incorporating negation bigrams as additional features into machine learning approaches. The experimental results show that considering sentiment shifting greatly improves the performance of term-counting approaches but only slightly improves the performance of machine learning approaches. Other studies such as Na et al. (2004), Ding et al. (2008), and Wilson et al. (2009) also explore negation shifting and achieve some improvements[1]. Nonetheless, as far as machine learning approaches are concerned, the improvement is rather insignificant (normally less than 1%). More recently, Ikeda et al. (2008) first propose a machine learning approach to detect polarity shifting for sentence-level sentiment classification, based on a manually-constructed dictionary containing thousands of positive and negative sentimental words, and then adopt a term-counting approach to incorporate polarity shifting information.

3 Sentiment Classification with Polarity Shifting Detection

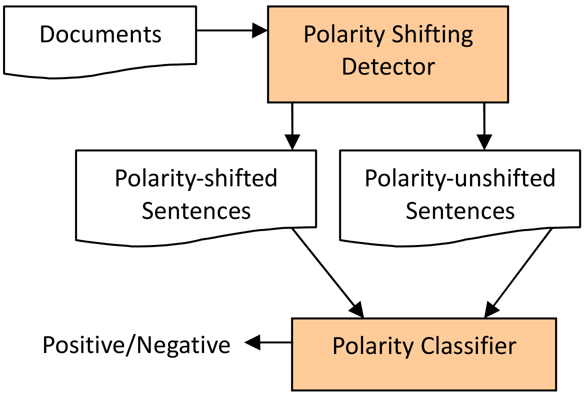


Figure 1: General framework of our approach

The motivation of our approach is to improve the performance of sentiment classification by robust treatment of sentiment polarity shifting between sentences. With the help of a binary classifier, the sentences in a document are divided into two parts: sentences which contain polarity shifting structures and sentences without any polarity shifting structure. Figure 1 illustrates the general framework of our approach. Note that this framework is a general one, that is, different polarity shifting detection methods can be applied to differentiate polarity-shifted sentences from those polarity-unshifted sentences and different polarity classification methods can be adopted to incorporate sentiment shifting information. For clarification, the training data used for polarity shifting detection and polarity classification are referred to as the polarity shifting training data and the polarity classification training data, respectively.

3.1 Polarity Shifting Detection

In this paper, polarity shifting means that the polarity of a sentence is different from the polarity expressed by the sum of the content words in the sentence. For example, in the sentence "*I am not disappointed*", the negation structure makes the polarity of the word '*disappointed*' different from that of the whole sentence (*negative* vs. *positive*). Apart from the negation structure, many other linguistic structures allow polarity shifting, such as contrast transition, modals, and pre-suppositional items (Polanyi and Zaenen, 2006). We refer these structures as polarity shifting structures.

One of the great challenges in building a polarity shifting detector lies on the lack of relevant training data since manually creating a large scale corpus of polarity shifting sentences is time-consuming and labor-intensive. Ikeda et al. (2008) propose an automatic way for collecting the polarity shifting training data based on a manually-constructed large-scale dictionary. Instead, we adopt a feature selection method to build a large scale training corpus of polarity shifting sentences, given only the already available document-level polarity classification training data. With the help of the feature selection method, the top-ranked word features with strong sentimental polarity orientation, e.g., '*great*', '*love*', '*worst*' are first chosen as the polarity trigger words. Then, those sentences with the top-ranked polarity trigger words in both categories of positive and negative documents are selected. Finally, those candidate sentences taking opposite-polarity compared to the containing trigger word are deemed as polarity-shifted.

The basic idea of automatically generating the polarity shifting training data is based on the assumption that the real polarity of a word or phrase is decided by the major polarity category where the word or phrase appears more often. As a result, the sentences in the

[1] Note that Ding et al. (2006) also consider *but*-clause, another important structure for sentiment shifting. Wilson et al. (2009) use conjunctive and dependency relations among polarity words.

frequently-occurring category would be seen as polarity-unshifted while the sentences in the infrequently-occurring category would be seen as polarity-shifted.

In the literature, various feature selection methods, such as Mutual Information (MI), Information Gain (IG) and Bi-Normal Separation (BNS) (Yang and Pedersen, 1997; Forman 2003), have been employed to cope with the problem of the high-dimensional feature space which is normal in sentiment classification.

In this paper, we employ the theoretical framework, proposed by Li et al. (2009), including two basic measurements, i.e. *frequency measurement* and *ratio measurement*, where the first measures, the document frequency of a term in one category, and the second measures, the ratio between the document frequency in one category and other categories. In particular, a novel method called Weighed Frequency and Odds (WFO) is proposed to incorporate both basic measurements:

$$WFO(t,c_i) = P(t|c_i)^{\lambda}\{\max(0,\log\frac{P(t|c_i)}{P(t|\overline{c_i})})\}^{1-\lambda}$$

where $P(t|c_i)$ denotes the probability that a document x contains the term t with the condition that x belongs to category c_i; $P(t|\overline{c_i})$ denotes the probability that a document x contains the term t with the condition that x does not belong to category c_i. The left part of the formula $P(t|c_i)$ implies the first basic measurement and the right part $\log(P(t|c_i)/P(t|\overline{c_i}))$ implies the second one. The parameter λ $(0 \le \lambda \le 1)$ is thus to tune the weight between the two basic measurements. Especially, when λ equals 0, the WFO method fades to the MI method which fully prefers the second basic measurement.

Figure 2 illustrates our algorithm for automatically generating the polarity shifting training data where c_1 and c_2 denote the two sentimental orientation categories, i.e. negative and positive. *Step A* segments a document into sentences with punctuations. Besides, two special words, '*but*' and '*and*', are used to further segment some contrast transition structures and compound sentences. *Step B* employs the WFO method to rank all features including the words. *Step D* extracts those polarity-shifted and polarity-unshifted sentences containing t_{top-i} where $N_{\max}$ denotes the upper-limit number of sentences in each category of the polarity shifting training data and *#(x)* denotes the total number of the elements in *x*. Apart from that, the first word in the following sentence is also included to capture a common kind of long-distance polarity shifting structure: contrast transition. Thus, important trigger words like '*however*' and '*but*' may be considered. Finally, *Step E* guarantees the balance between the two categories of the polarity shifting training data.

Given the polarity shifting training data, we apply SVM classification algorithm to train a polarity-shifting detector with word unigram features.

Input:

The polarity classification training data: the negative sentimental document set D_{c_1} and the positive sentimental document set D_{c_2}.

Output:

The polarity shifting training data: the polarity-unshifted sentence set $S_{unshift}$ and the polarity-shifted sentence set S_{shift}.

Procedure:

A. *Segment documents* D_{c_1} *and* D_{c_2} *to single sentences* S_{c_1} *and* S_{c_2}.

B. *Apply feature selection on the polarity classification training data and get the ranked features,* $(t_{top-1},...,t_{top-i},...,t_{top-N})$

C. $S_{shift} = \{\}$, $S_{unshift} = \{\}$

D. *For* t_{top-i} *in* $(t_{top-1},...,t_{top-i},...,t_{top-N})$ *:*

D1) if #(S_{shift})> $N_{\max}$ *: break*

D2) Collect all sentences S_{top-i,c_1} *and* S_{top-i,c_2} *which contain* t_{top-i} *from* S_{c_1} *and* S_{c_2} *respectively*

D3) if #(S_{top-i,c_1})>#(S_{top-i,c_2}):

put S_{top-i,c_2} *into* S_{shift}

put S_{top-i,c_1} *into* $S_{unshift}$

else:

put S_{top-i,c_1} *into* S_{shift}

put S_{top-i,c_2} *into* $S_{unshift}$

E. *Randomly select* $N_{\max}$ *sentences from* $S_{unshift}$ *as the output of* $S_{unshift}$

Figure 2: The algorithm for automatically generating the polarity shifting training data

3.2 Polarity Classification with Classifier Combination

After polarity shifting detection, each document in the polarity classification training data is divided into two parts, one containing polarity-shifted sentences and the other containing polarity-unshifted sentences, which are used to form the polarity-shifted training data and the polarity-unshifted training data. In this way, two different polarity classifiers, f_1 and f_2, can be trained on the polarity-shifted training data and the polarity-unshifted training data respectively. Along with classifier f_3, trained on all original polarity classification training data, we now have three base classifiers in hand for possible classifier combination via a multiple classifier system.

The key issue in constructing a multiple classifier system (MCS) is to find a suitable way to combine the outputs of the base classifiers. In MCS literature, various methods are available for combining the outputs, such as fixed rules including the voting rule, the product rule and the sum rule (Kittler et al., 1998) and trained rules including the weighted sum rule (Fumera and Roli, 2005) and the meta-learning approaches (Vilalta and Drissi, 2002). In this study, we employ the product rule, a popular fixed rule, and stacking (Džeroski and Ženko, 2004), a well-known trained rule, to combine the outputs.

Formally, each base classifier provides some kind of confidence measurements, e.g., posterior probabilities of the test sample belonging to each class. Formally, each base classifier f_l $(l = 1,2,3)$ assigns a test sample (denoted as x_l) a posterior probability vector $\vec{P}(x_l)$:

$$\vec{P}(x_l) = (p(c_1 \mid x_l), p(c_2 \mid x_l))^t$$

where $p(c_1 \mid x_l)$ denotes the probability that the l-th base classifier considers the sample belonging c_1.

The product rule combines the base classifiers by multiplying the posterior possibilities and using the multiplied possibility for decision, i.e.

$$assign \quad y \rightarrow c_j \;\; when\; j = \arg\max_i \prod_{l=1}^{3} p(c_i \mid x_l)$$

Stacking belongs to well-known meta-learning (Vilalta and Drissi, 2002). The key idea behind meta-learning is to train a meta-classifier with input attributes that are the outputs of the base classifiers. Hence, meta-learning usually needs some development data for generating the meta-training data. Let x' denote a feature vector of a sample from the development data. The output of the l-th base classifier f_l on this sample is the probability distribution over the category set $\{c_1, c_2\}$, i.e.

$$\vec{P}(x'_l) = (p(c_1 \mid x'_l), p_l(c_2 \mid x'_l))$$

A meta-classifier can be trained using the development data with the meta-level feature vector $x^{meta} \in R^{2\times3}$

$$x^{meta} = (\vec{P}(x'_{l=1}), \vec{P}(x'_{l=2}), \vec{P}(x'_{l=3}))$$

Stacking is a specific meta-learning rule, in which a leave-one-out or a cross-validation procedure on the training data is applied to generate the meta-training data instead of using extra development data. In our experiments, we perform stacking with 10-fold cross-validation to generate the meta-training data.

4 Experimentation

4.1 Experimental Setting

The experiments are carried out on product reviews from four domains: books, DVDs, electronics, and kitchen appliances (Blitzer et al., 2007)[2]. Each domain contains 1000 positive and 1000 negative reviews.

For sentiment classification, all classifiers including the polarity shifting detector, three base classifiers and the meta-classifier in stacking are trained by SVM using the SVM-light tool [3] with Logistic Regression method for probability measuring (Platt, 1999).

In all the experiments, each dataset is randomly and evenly split into two subsets: 50% documents as the training data and the remaining 50% as the test data. The features include word unigrams and bigrams with Boolean weights.

4.2 Experimental Results on Polarity Shifting Data

To better understand the polarity shifting phenomena in document-level sentiment classification, we randomly investigate 200

[2] This data set is collected by Blitzer et al. (2007): http://www.seas.upenn.edu/~mdredze/datasets/sentiment/

[3] It is available at: http://svmlight.joachims.org/

polarity-shifted sentences, together with their contexts (i.e. the sentences before and after it), automatically generated by the WFO ($\lambda = 0$) feature selection method. We find that nearly half of the automatically generated polarity-shifted sentences are actually polarity-unshifted sentences or difficult to decide. That is to say, the polarity shifting training data is noisy to some extent. One main reason is that some automatically selected trigger words do not really contain sentiment information, e.g., '*hear*', '*information*' etc. Another reason is that some reversed opinion is given in a review without any explicit polarity shifting structures.

To gain more insights, we manually checked 100 sentences which are explicitly polarity-shifted and can also be judged by human according to their contexts. Table 1 presents some typical structures causing polarity shifting. It shows that the most common polarity shifting type is Explicit Negation (37%), usually expressed by trigger words such as '*not*', '*no*', or '*without*', e.g., in the sentence '*I am not happy with this flashcard at all*'. Another common type of polarity shifting is Contrast Transition (20%), expressed by trigger words such as '*however*', e.g., in the sentence '*It is large and stylish, however, I cannot recommend it because of the lid*'. Other less common yet productive polarity shifting types include Exception and Until. Exception structure is usually expressed by the trigger phrase '*the only*' to indicate the one and only advantage of the product, e.g., in the sentence '*The only thing that I like about it is that bamboo is a renewable resource*'. Until structure is often expressed by the trigger word '*until*' to show the reversed polarity, e.g. in the sentence '*This unit was a great addition until the probe went bad after only a few months*'.

Polarity Shifting Structures	Trigger Words/Phrases	Distribution (%)
Explicit Negation	*not, no, without*	37
Contrast Transition	*but, however, unfortunately*	20
Implicit Negation	*avoid, hardly,*	7
False Impression	*look, seem*	6
Likelihood	*probably, perhaps*	5
Counter-factual	*should, would*	5
Exception	*the only*	5
Until	*until*	3

Table 1: Statistics on various polarity shifting structures

4.3 Experimental Results on Polarity Classification

For comparison, several classifiers with different classification methods are developed.

1) Baseline classifier, which applies SVM with all unigrams and bigrams. Note that it also serves as a base classifier in the following combined classifiers.

2) Base classifier 1, a base classifier for the classifier combination method. It works on the polarity-unshifted data.

3) Base classifier 2, another base classifier for the classifier combination method. It works on the polarity-shifted data.

4) Negation classifier, which applies SVM with all unigrams and bigrams plus negation bigrams. It is a natural extension of the baseline classifier with the consideration of negation bigrams. In this study, the negation bigrams are collected using some negation trigger words, such as '*not*' and '*never*'. If a negation trigger word is found in a sentence, each word in the sentence is attached with the word '*_not*' to form a negation bigram.

5) Product classifier, which combines the baseline classifier, the base classifier 1 and the base classifier 2 using the product rule.

6) Stacking classifier, a combined classifier similar to the **Product classifier**. It uses the stacking classifier combination method instead of the product rule.

Please note that we do not compare our approach with the one as proposed in Ikeda et al. (2008) due to the absence of a manually-collected sentiment dictionary. Besides, it is well known that a combination strategy itself is capable of improving the classification performance. To justify whether the improvement is due to the combination strategy or our polarity shifting detection or both, we first randomly split the training data into two portions and train two base classifiers on each portion, then apply the stacking method to combine them along with the baseline classifier. The corresponding results are shown as 'Random+Stacking' in Table 2. Finally, in our experiments, *t*-test is performed to evaluate the significance of the performance improvement between two systems employing different methods (Yang and Liu, 1999).

Domain	Baseline	Base Classifier 1	Base Classifier 2	Negation Classifier	Random + Stacking	Shifting + Product	Shifting + Stacking
Book	0.755	0.756	0.670	0.759	0.764	0.772	**0.785**
DVD	0.750	0.743	0.667	0.748	0.759	0.768	**0.770**
Electronic	0.779	0.786	0.711	0.785	0.789	0.820	**0.830**
Kitchen	0.818	0.814	0.683	0.826	0.835	0.840	**0.849**

Table 2: Performance comparison of different classifiers with equally-splitting between training and test data

Performance comparison of different classifiers

Table 2 shows the accuracy results of different methods using 2000 polarity shifted sentences and 2000 polarity-unshifted sentences to train the polarity shifting detector (N_{max}=2000). Compared to the baseline classifier, it shows that: 1) The base classifier 1, which only uses the polarity-unshifted sentences as the training data, achieves similar performance. 2) The base classifier 2 achieves much lower performance due to much fewer sentences involved. 3) Including negation bigrams usually allows insignificant improvements (*p-value*>0.1), which is consistent with most of previous works (Pang et al., 2002; Kennedy and Inkpen, 2006). 4) Both the product and stacking classifiers with polarity shifting detection significantly improve the performance (*p-value*<0.05). Compared to the product rule, the stacking classifier is preferable, probably due to the performance unbalance among the individual classifiers, e.g., the performance of the base classifier 2 is much lower than the other two. Although stacking with two randomly generated base classifiers, i.e. "Random + Stacking", also consistently outperforms the baseline classifier, the improvements are much lower than what has been achieved by our approach. This suggests that both the classifier combination strategy and polarity shifting detection contribute to the overall performance improvement.

Effect of WFO feature selection method

Figure 3 presents the accuracy curve of the stacking classifier when using different Lambda (λ) values in the WFO feature selection method. It shows that those feature selection methods which prefer frequency information, e.g., MI and BNS, are better in automatically generating the polarity shifting training data. This is reasonable since high frequency terms, e.g., '*is*', '*it*', '*a*', etc., tend to obey our assumption that the real polarity of one top term should belong to the polarity category where the term appears frequently.

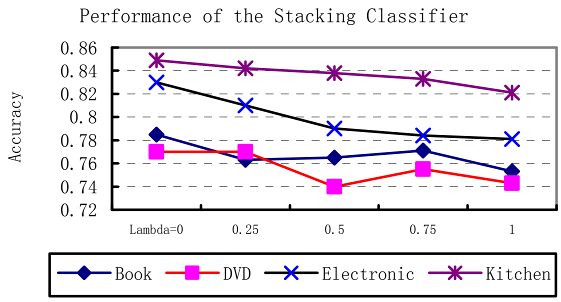


Figure 3: Performance of the stacking classifier using WFO with different Lambda (λ) values

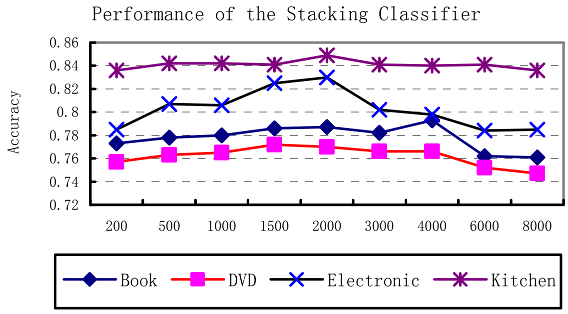


Figure 4: Performance of the stacking classifier over different sizes of the polarity shifting training data (with N_{max} sentences in each category)

Effect of a classifier over different sizes of the polarity shifting training data

Another factor which might influence the overall performance is the size of the polarity shifting training data. Figure 4 presents the overall performance on different numbers of the polarity shifting sentences when using the stacking classifier. It shows that 1000 to 4000 sentences are enough for the performance improvement. When the number is too large, the noisy training data may harm polarity shifting detection. When the number is too small, it is not enough for the automatically generated polarity shifting training data to capture various polarity shifting structures.

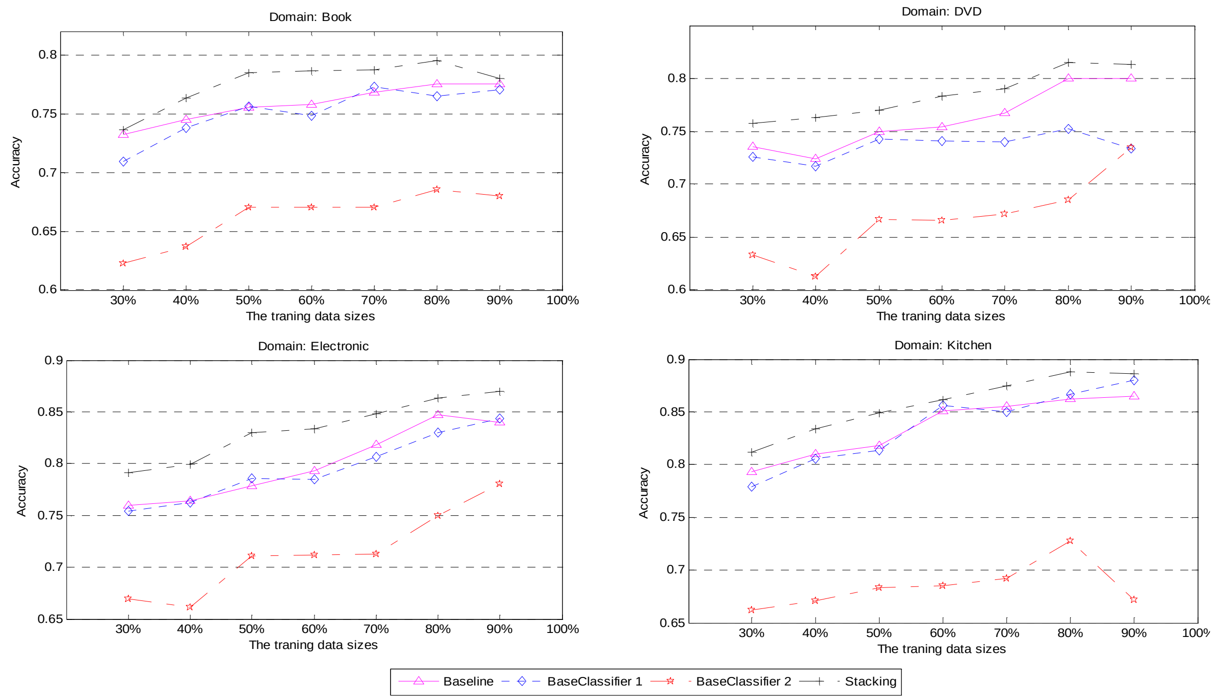


Figure 5: Performance of different classifiers over different sizes of the polarity classification training data

Effect of different classifiers over different sizes of the polarity classification training data

Figure 5 shows the classification results of different classifiers with varying sizes of the polarity classification training data. It shows that our approach is able to improve the overall performance robustly. We also notice the big difference between the performance of the baseline classifier and that of the base classifier 1 when using 30% training data in Book domain and 90% training data in DVD domain. Detailed exploration of the polarity shifting sentences in the training data shows that this difference is mainly attributed to the poor performance of the polarity shifting detector. Even so, the stacking classifier guarantees no worse performance than the baseline classifier.

5 Conclusion and Future Work

In this paper, we propose a novel approach to incorporate polarity shifting information into document-level sentiment classification. In our approach, we first propose a machine-learning-based classifier to detect polarity shifting and then apply two classifier combination methods to perform polarity classification. Particularly, the polarity shifting training data is automatically generated through a feature selection method. As shown in our experimental results, our approach is able to consistently improve the overall performance across different domains and training data sizes, although the automatically generated polarity shifting training data is prone to noise. Furthermore, we conclude that those feature selection methods, which prefer frequency information, e.g., MI and BNS, are good choices for generating the polarity shifting training data.

In our future work, we will explore better ways in generating less-noisy polarity shifting training data. In addition, since our approach is language-independent, it is readily applicable to sentiment classification tasks in other languages.

For availability of the automatically generated polarity shifting training data, please contact the first author (for research purpose only).

Acknowledgments

This research work has been partially supported by Start-up Grant for Newly Appointed Professors, No. 1-BBZM in the Hong Kong Polytechnic University and two NSFC grants, No. 60873150 and No. 90920004. We also thank the three anonymous reviewers for their helpful comments.

References

Blitzer J., M. Dredze, and F. Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of ACL-07*.

Dasgupta S. and V. Ng. 2009. Mine the Easy and Classify the Hard: Experiments with Automatic Sentiment Classification. In *Proceedings of ACL-IJCNLP-09*.

Ding X., B. Liu, and P. Yu. 2008. A Holistic Lexicon-based Approach to Opinion Mining. In *Proceedings of the International Conference on Web Search and Web Data Mining, WSDM-08*.

Džeroski S. and B. Ženko. 2004. Is Combining Classifiers with Stacking Better than Selecting the Best One? *Machine Learning*, vol.54(3), pp.255-273, 2004.

Forman G. 2003. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *The Journal of Machine Learning Research*, 3(1), pp.1289-1305.

Fumera G. and F. Roli. 2005. A Theoretical and Experimental Analysis of Linear Combiners for Multiple Classifier Systems. *IEEE Trans. PAMI*, vol.27, pp.942–956, 2005

Ikeda D., H. Takamura, L. Ratinov, and M. Okumura. 2008. Learning to Shift the Polarity of Words for Sentiment Classification. In *Proceedings of IJCNLP-08*.

Kennedy, A. and D. Inkpen. 2006. Sentiment Classification of Movie Reviews using Contextual Valence Shifters. *Computational Intelligence*, vol.22(2), pp.110-125, 2006.

Kim S. and E. Hovy. 2004. Determining the Sentiment of Opinions. In *Proceedings of COLING-04*.

Kittler J., M. Hatef, R. Duin, and J. Matas. 1998. On Combining Classifiers. *IEEE Trans. PAMI*, vol.20, pp.226-239, 1998

Li S., R. Xia, C. Zong, and C. Huang. 2009. A Framework of Feature Selection Methods for Text Categorization. In *Proceedings of ACL-IJCNLP-09*.

Li S. and C. Zong. 2008. Multi-domain Sentiment Classification. In *Proceedings of ACL-08: HLT*, short paper.

Liu B., M. Hu, and J. Cheng. 2005. Opinion Observer: Analyzing and Comparing Opinions on the Web. In *Proceedings of WWW-05*.

Na J., H. Sui, C. Khoo, S. Chan, and Y. Zhou. 2004. Effectiveness of Simple Linguistic Processing in Automatic Sentiment Classification of Product Reviews. In *Conference of the International Society for Knowledge Organization (ISKO-04)*.

Pang B. and L. Lee. 2004. A Sentimental Education: Sentiment Analysis using Subjectivity Summarization based on Minimum Cuts. In *Proceedings of ACL-04*.

Pang B., L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment Classification using Machine Learning Techniques. In *Proceedings of EMNLP-02*.

Platt J. 1999. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. In: A. Smola, P. Bartlett, B. Schoelkopf and D. Schuurmans (Eds.): *Advances in Large Margin Classiers*. MIT Press, Cambridge, 61﹣74.

Polanyi L. and A. Zaenen. 2006. Contextual Valence Shifters. *Computing attitude and affect in text: Theory and application*. Springer Verlag.

Riloff E., S. Patwardhan, and J. Wiebe. 2006. Feature Subsumption for Opinion Analysis. In *Proceedings of EMNLP-06*.

Turney P. 2002. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of ACL-02*.

Vilalta R. and Y. Drissi. 2002. A Perspective View and Survey of Meta-learning. *Artificial Intelligence Review*, 18(2), pp. 77–95.

Wan X. 2009. Co-Training for Cross-Lingual Sentiment Classification. In *Proceedings of ACL-IJCNLP-09*.

Wiebe J. 2000. Learning Subjective Adjectives from Corpora. In *Proceedings of AAAI-2000*.

Wilson T., J. Wiebe, and P. Hoffmann. 2009. Recognizing Contextual Polarity: An Exploration of Features for Phrase-Level Sentiment Analysis. *Computational Linguistics*, vol.35(3), pp.399-433, 2009.

Yang Y. and X. Liu, X. 1999. A Re-Examination of Text Categorization methods. In *Proceedings of SIGIR-99*.

Yang Y. and J. Pedersen. 1997. A Comparative Study on Feature Selection in Text Categorization. In *Proceedings of ICML-97*.

Improving Corpus Comparability for Bilingual Lexicon Extraction from Comparable Corpora

Bo Li, Eric Gaussier
Laboratoire d'Informatique de Grenoble (LIG)
Université de Grenoble
firstname.lastname@imag.fr

Abstract

Previous work on bilingual lexicon extraction from comparable corpora aimed at finding a good representation for the usage patterns of source and target words and at comparing these patterns efficiently. In this paper, we try to work it out in another way: improving the quality of the comparable corpus from which the bilingual lexicon has to be extracted. To do so, we propose a measure of comparability and a strategy to improve the quality of a given corpus through an iterative construction process. Our approach, being general, can be used with any existing bilingual lexicon extraction method. We show here that it leads to a significant improvement over standard bilingual lexicon extraction methods.

1 Introduction

Bilingual dictionaries are an essential resource in many multilingual natural language processing (NLP) tasks such as machine translation (Och and Ney, 2003) and cross-language information retrieval (CLIR) (Ballesteros and Croft, 1997). Hand-coded dictionaries are of high quality, but expensive to build and researchers have tried, since the end of the 1980s, to automatically extract bilingual lexicons from parallel corpora (see (Chen, 1993; Kay and Röscheisen, 1993; Melamed, 1997a; Melamed, 1997b) for early work). Parallel corpora are however difficult to get at in several domains, and the majority of bilingual collections are comparable and not parallel. Due to their low cost of acquisition, several researchers have tried to exploit such comparable corpora for bilingual lexicon extraction (Fung and McKeown, 1997; Fung and Yee, 1998; Rapp, 1999; Déjean et al., 2002; Gaussier et al., 2004; Robitaille et al., 2006; Morin et al., 2007; Yu and Tsujii, 2009). The notion of comparability is however a loose one, and comparable corpora range from lowly comparable ones to highly comparable ones and parallel ones. For data-driven NLP techniques, using better corpora often leads to better results, a fact which should be true for the task of bilingual lexicon extraction. This point has largely been ignored in previous work on the subject. In this paper, we develop a well-founded strategy to improve the quality of a comparable corpus, so as to improve in turn the quality of the bilingual lexicon extracted. To do so, we first propose a measure of comparability which we then use in a method to improve the quality of the existing corpus.

The remainder of the paper is organized as follows: Section 2 introduces the experimental materials used for the different evaluations; comparability measures are then presented and evaluated in Section 3; in Section 4, we detail and evaluate a strategy to improve the quality of a given corpus while preserving its vocabulary; the method used for bilingual lexicon extraction is then described and evaluated in Section 5. Section 6 is then devoted to a discussion, prior to the conclusion given in Section 7.

2 Experimental Materials

For the experiments reported here, several corpora were used: the parallel English-French *Europarl* corpus (Koehn, 2005), the TREC

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 644–652,
Beijing, August 2010

(http://trec.nist.gov/) *Associated Press* corpus (*AP*, English) and the corpora used in the multilingual track of CLEF (http://www.clef-campaign.org) which includes the *Los Angeles Times* (*LAT94*, English), *Glasgow Herald* (*GH95*, English), *Le Monde* (*MON94*, French), *SDA French 94* (*SDA94*, French) and *SDA French 95* (*SDA95*, French). In addition to these existing corpora, two monolingual corpora from the Wikipedia dump[1] were built. For English, all the articles below the root category *Society* with a depth less than 4[2] were retained. For French, all the articles with a depth less than 7 below the category *Société* are extracted. As a result, the English corpus *Wiki-En* consists of 367,918 documents and the French one *Wiki-Fr* consists of 378,297 documents.

The bilingual dictionary used in our experiments is constructed from an online dictionary. It consists of 33,372 distinct English words and 27,733 distinct French words, which constitutes 75,845 translation pairs. Standard preprocessing steps: tokenization, POS-tagging and lemmatization are performed on all the linguistic resources. We will directly work on lemmatized forms of content words (nouns, verbs, adjectives, adverbs).

3 Measuring Comparability

As far as we can tell, there are no practical measures with which we can judge the degree of comparability of a bilingual corpus. In this paper, we propose a comparability measure based on the expectation of finding the translation for each word in the corpus. The measure is light-weighted and does not depend on complex resources like the machine translation system. For convenience, the following discussions will be made in the context of the English-French comparable corpus.

3.1 The Comparability Measure

For the comparable corpus $\mathcal{C}$, if we consider the translation process from the English part $\mathcal{C}_e$ to the French part $\mathcal{C}_f$, a comparability measure M_{ef} can be defined on the basis of the expectation of finding, for each English word w_e in the vocabulary $\mathcal{C}_e^v$ of $\mathcal{C}_e$, its translation in the vocabulary $\mathcal{C}_f^v$ of $\mathcal{C}_f$. Let σ be a function indicating whether a translation from the translation set $\mathcal{T}_w$ of w is found in the vocabulary $\mathcal{C}^v$ of a corpus $\mathcal{C}$, i.e.:

$$\sigma(w, \mathcal{C}^v) = \begin{cases} 1 & \text{iff } \mathcal{T}_w \cap \mathcal{C}^v \neq \emptyset \\ 0 & \text{else} \end{cases}$$

M_{ef} is then defined as:

$$\begin{aligned} & M_{ef}(\mathcal{C}_e, \mathcal{C}_f) = \mathbb{E}(\sigma(w, \mathcal{C}_f^v) | w \in \mathcal{C}_e^v) \\ & = \sum_{w \in \mathcal{C}_e^v} \underbrace{\sigma(w, \mathcal{C}_f^v) \cdot Pr(w \in \mathcal{C}_e^v)}_{A_w} \\ & = \frac{|\mathcal{C}_e^v|}{|\mathcal{C}_e^v \cap \mathcal{D}_e^v|} \sum_{w \in \mathcal{C}_e^v \cap \mathcal{D}_e^v} A_w \end{aligned}$$

where $\mathcal{D}_e^v$ is the English part of a given, independent bilingual dictionary $\mathcal{D}$, and where the last equality is based on the fact that, the comparable corpus and the bilingual dictionary being independent of one another, the probability of finding the translation in $\mathcal{C}_f^v$ of a word w is the same for w is in $\mathcal{C}_e^v \cap \mathcal{D}_e^v$ and in $\mathcal{C}_e^v \backslash \mathcal{D}_e^v$[3]. Furthermore, the presence of common words suggests that one should rely on a presence/absence criterion rather than on the number of occurrences to avoid a bias towards common words. Given the natural language text, our evaluation will show that the simple presence/absence criterion can perform very well. This leads to $Pr(w \in \mathcal{C}_e^v) = 1/|\mathcal{C}_e^v|$, and finally to:

$$M_{ef}(\mathcal{C}_e, \mathcal{C}_f) = \frac{1}{|\mathcal{C}_e^v \cap \mathcal{D}_e^v|} \sum_{w \in \mathcal{C}_e^v \cap \mathcal{D}_e^v} \sigma(w, \mathcal{C}_f^v)$$

This formula shows that M_{ef} is actually the proportion of English words translated in the French part of the comparable corpus. Similarly, the counterpart of M_{ef}, M_{fe}, is defined as:

$$M_{fe}(\mathcal{C}_e, \mathcal{C}_f) = \frac{1}{|\mathcal{C}_f^v \cap \mathcal{D}_f^v|} \sum_{w \in \mathcal{C}_f^v \cap \mathcal{D}_f^v} \sigma(w, \mathcal{C}_e^v)$$

[1]The Wikipedia dump files can be downloaded at http://download.wikimedia.org. In this paper, we use the English dump file on July 13, 2009 and the French dump file on July 7, 2009.

[2]There are several cycles in the category tree of Wikipedia. It is thus necessary to define a threshold on the depth to make the iterative process feasible.

[3]The fact can be reliable only when a substantial part of the corpus vocabulary is covered by the dictionary. Fortunately, the constraint is satisfied in most applications where the common but not the specialized corpora like the medical corpora are involved.

and measures the proportion of French words in $\mathcal{C}_f^v$ translated in the English part of the comparable corpus. A symmetric version of these measures is obtained by considering the proportion of the words (both English and French) for which a translation can be found in the corpus:

$$M(\mathcal{C}_e, \mathcal{C}_f) = \frac{\sum_{w \in \mathcal{C}_e^v \cap \mathcal{D}_e^v} \sigma(w, \mathcal{C}_f^v) + \sum_{w \in \mathcal{C}_f^v \cap \mathcal{D}_f^v} \sigma(w, \mathcal{C}_e^v)}{|\mathcal{C}_e^v \cap \mathcal{D}_e^v| + |\mathcal{C}_f^v \cap \mathcal{D}_f^v|}$$

We now present an evaluation of these measures on artificial test corpora.

3.2 Validation

In order to test the comparability measures, we developed gold-standard comparability scores from the *Europarl* and *AP* corpora. We start from the parallel corpus, *Europarl*, of which we degrade the comparability by gradually importing some documents from either *Europarl* or *AP*. Three groups (G_a, G_b, G_c) of comparable corpora are built in this fashion. Each group consists of test corpora with a gold-standard comparability ranging, arbitrarily, from 0 to 1 and corresponding to the proportion of documents in "parallel" translation. The first group G_a is built from *Europarl* only. First, the *Europarl* corpus is split into 10 equal parts, leading to 10 parallel corpora (P_1, P_2, $\ldots$, P_{10}) with a gold-standard comparability arbitrarily set to 1. Then for each parallel corpus, e.g. P_i, we replace a certain proportion p of the English part with documents of the same size from another parallel corpus $P_j (j \neq i)$, producing the new corpus P_i' with less comparability which is the gold-standard comparability $1 - p$. For each P_i, as p increases, we obtain several comparable corpora with a decreasing gold-standard comparability score. All the P_i and their descendant corpora constitute the group G_a. The only difference between G_b and G_a is that, in G_b, the replacement in P_i is done with documents from the *AP* corpus and not from *Europarl*. In G_c, we start with 10 final, comparable corpora P_i' from G_a. These corpora have a gold-standard comparability of 0 in G_a, and of 1 in G_c. Then each P_i' is further degraded by replacing certain portions with documents from the *AP* corpus.

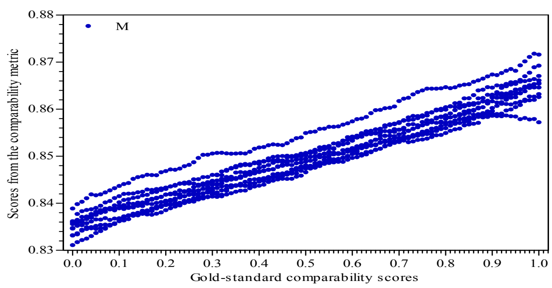


Figure 1: Evolution of M wrt gold-standard on the corpus group G_c (x-axis: gold-standard scores, y-axis: *M* scores)

We then computed, for each comparable corpus in each group, its comparability according to one of the comparability measures. Figure 1 plots the measure M for ten comparable corpora and their descendants from G_c, according to their gold-standard comparability scores. As one can note, the measure M is able to capture almost all the differences in comparability and is strongly correlated with the gold-standard. The correlation between the different measures and the gold-standard is finally computed with Pearson correlation coefficient. The results obtained are listed in Table 1. As one can note, M_{fe} performs worst among the three measures, the reason being that the process to construct G_b and G_c yields unbalanced bilingual corpora, the English section being larger than the French one. Translations of French words are still likely to be found in the English corpus, even though the corpora are not comparable. On all the 3 groups, M performs best and correlates very well with the gold standard, meaning that M was able to capture all the differences in comparability artificially introduced in the degradation process we have considered. This is the measure we will retain in the following parts.

	M_{ef}	M_{fe}	M
G_a	0.897	0.770	0.936
G_b	0.955	0.190	0.979
G_c	0.940	-0.595	0.960

Table 1: Correlation scores for the different comparability measures on the 3 groups of corpora

Having established a measure for the degree of comparability of bilingual corpora, we now turn to the problem of improving the quality of comparable corpora.

4 Improving Corpus Quality

We here try to improve the quality of a given corpus $\mathcal{C}$, which we will refer to as the *base corpus*, by extracting the highly comparable subpart $\mathcal{C}_H$ which is above a certain degree of comparability η (Step 1), and by enriching the lowly comparable part $\mathcal{C}_L$ with texts from other sources (Step 2). As we are interested in extracting information related to the vocabulary of the base corpus, we want the newly built corpus to contain a substantial part of the base corpus. This can be achieved by preserving in Step 1 as many documents from the base corpus as possible (e.g. by considering low values of η), and by using in step 2 sources close to the base corpus.

4.1 Step 1: Extracting $\mathcal{C}_H$

The strategy consisting of building all the possible sub-corpora of a given size from a given comparable corpora is not realistic as soon as the number of documents making up the corpora is larger than a few thousands. In such cases, better ways for extracting subparts have to be designed. The strategy we have adopted here aims at efficiently extracting a subpart of $\mathcal{C}$ above a certain degree of comparability and is based on the following property.

Property 1. *Let d_e^1 and d_e^2 (resp. d_f^1 and d_f^2) be two English (resp. French) documents from a bilingual corpus $\mathcal{C}$. We consider, as before, that the bilingual dictionary $\mathcal{D}$ is independent from $\mathcal{C}$. Let $(d_e^{1'}, d_f^{1'})$ be such that: $d_e^{1'} \subseteq d_e^1$, $d_f^{1'} \subseteq d_f^1$, which means $d_e^{1'}$ is a subpart of d_e^1 and $d_f^{1'}$ is a subpart of d_f^1.*
We assume:

(i) $\frac{|d_e^1 \cup d_e^2|}{|d_e^2|} = \frac{|d_f^1 \cup d_f^2|}{|d_f^2|}$

(ii) $M_{ef}(d_e^{1'}, d_f^1) \geq M_{ef}(d_e^2, d_f^2)$
$M_{fe}(d_e^1, d_f^{1'}) \geq M_{fe}(d_e^2, d_f^2)$

Then:

$$M(d_e^2, d_f^2) \leq M(d_e^1 \cup d_e^2, d_f^1 \cup d_f^2)$$

Proof [sketch]: Let $B = (d_e^1 \cup d_e^2) \cap \mathcal{D}_e^v) \backslash (d_e^2 \cap \mathcal{D}_e^v)$. One can show, by exploiting condition (ii), that:

$$\sum_{w \in B} \sigma(w, d_f^1 \cup d_f^2) \geq |B| M_{ef}(d_e^2, d_f^2)$$

and similarly that:

$$\sum_{w \in d_e^2 \cap \mathcal{D}_e^v} \sigma(w, d_f^1 \cup d_f^2) \geq |d_e^2 \cap \mathcal{D}_e^v| M_{ef}(d_e^2, d_f^2)$$

Then exploiting condition (i), and the independence between the corpus and the dictionary, one arrives at:

$$\frac{\sum_{w \in (d_e^1 \cup d_e^2) \cap \mathcal{D}_e^v} \sigma(w, d_f^1 \cup d_f^2)}{|(d_e^1 \cup d_e^2) \cap \mathcal{D}_e^v| + |(d_f^1 \cup d_f^2) \cap \mathcal{D}_f^v|} \geq \frac{|d_e^2 \cap \mathcal{D}_e^v| M_{ef}(d_e^2, d_f^2)}{|d_e^2 \cap \mathcal{D}_e^v| + |d_f^2 \cap \mathcal{D}_f^v|}$$

The same development on M_{fe} completes the proof. □

Property 1 shows that one can incrementally extract from a bilingual corpus a subpart with a guaranteed minimum degree of comparability η by iteratively adding new elements, provided (a) that the new elements have a degree of comparability of at least η and (b) that they are less comparable than the currently extracted subpart (conditions (ii)). This strategy is described in Algorithm 1. Since the degree of comparability is always above a certain threshold and since the new documents selected (d_e^2, d_f^2) are the most comparable among the remaining documents, condition (i) is likely to be satisfied, as this condition states that the increase in the vocabulary from the second documents to the union of the two is the same in both languages. Similarly, considering new elements by decreasing comparability scores is a necessary step for the satisfaction of condition (ii), which states that the current subpart should be uniformly more comparable than the element to be added. Hence, the conditions for property 1 to hold are met in Algorithm 1, which finally yields a corpus with a degree of comparability of at least η.

4.2 Step 2: Enriching $\mathcal{C}_L$

This step tries to absorb knowledge from other resources, which will be called *external corpus*,

Algorithm 1

Input:

English document set $\mathcal{C}_e^d$ of $\mathcal{C}$

French document set $\mathcal{C}_f^d$ of $\mathcal{C}$

Threshold η

Output:

$\mathcal{C}_H$, consisting of the English document set $\mathcal{S}_e$ and the French document set $\mathcal{S}_f$

Initialize $\mathcal{S}_e = \emptyset, \mathcal{S}_f = \emptyset$, temp $= 0$;

repeat

$\quad (d_e, d_f) = \underset{d_e \in \mathcal{C}_e^d, d_f \in \mathcal{C}_f^d}{\arg\max} M(d_e, d_f)$;

$\quad \text{temp} = \underset{d_e \in \mathcal{C}_e^d, d_f \in \mathcal{C}_f^d}{\max} M(d_e, d_f)$;

$\quad$ **if** temp $\geq \eta$ **then**

$\quad\quad$ Add d_e into $\mathcal{S}_e$ and add d_f into $\mathcal{S}_f$;

$\quad\quad \mathcal{C}_e^d = \mathcal{C}_e^d \backslash d_e$, $\mathcal{C}_f^d = \mathcal{C}_f^d \backslash d_f$;

$\quad$ **end if**

until $\mathcal{C}_e^d = \emptyset$ or $\mathcal{C}_f^d = \emptyset$ or temp $< \eta$

return $\mathcal{C}_H$;

to enrich the lowly comparable part $\mathcal{C}_L$ which is the left part in $\mathcal{C}$ during the creation of $\mathcal{C}_H$. One choice for obtaining the external corpus $\mathcal{C}_T$ is to fetch documents which are likely to be comparable from the Internet. In this case, we first extract representative words for each document in $\mathcal{C}_L$, translate them using the bilingual dictionary and retrieve associated documents via a search engine. An alternative approach is of course to use existing bilingual corpora. Once $\mathcal{C}_T$ has been constructed, the lowly comparable part $\mathcal{C}_L$ can be enriched in exactly the same way as in section 4.1: First, Algorithm 1 is used on the English part of $\mathcal{C}_L$ and the French part of $\mathcal{C}_T$ to get the high-quality document pairs. Then the French part of $\mathcal{C}_L$ is enriched with the English part of $\mathcal{C}_T$ by the same algorithm. All the high-quality document pairs are then added to $\mathcal{C}_H$ to constitute the final result.

4.3 Validation

We use here *GH95* and *SDA95* as the base corpus $\mathcal{C}^0$. In order to illustrate that the efficiency of the proposed algorithm is not confined to a specific external resource, we consider two external resources: (a) $\mathcal{C}_T^1$ made of *LAT94*, *MON94* and *SDA94*, and (b) $\mathcal{C}_T^2$ consisting of *Wiki-En* and *Wiki-Fr*. The number of documents in all the corpora after elimination of short documents ($<$ 30 words) is listed in Table 2.

	$\mathcal{C}^0$	$\mathcal{C}_T^1$	$\mathcal{C}_T^2$
English	55,989	109,476	367,918
French	42,463	87,086	378,297

Table 2: The size of the corpora in the experiments

For the extraction of the highly comparable part $\mathcal{C}_H$ from the base corpus $\mathcal{C}^0$, we set η to 0.3 so as to extract a substantial subpart of $\mathcal{C}^0$. After this step, corresponding to Algorithm 1, we have 20,124 English-French document pairs in $\mathcal{C}_H$. The second step is to enrich the lowly comparable part $\mathcal{C}_L$ of the base corpus documents from the external resources $\mathcal{C}_T^1$ and $\mathcal{C}_T^2$. The final corpora we obtain consist of 46,996 document pairs for $\mathcal{C}^1$ (with $\mathcal{C}_T^1$) and of 54,402 document pairs for $\mathcal{C}^2$ (with $\mathcal{C}_T^2$), size similar to the one of $\mathcal{C}^0$. The proportion of documents (columns "D-e" and "D-f"), sentences (columns "S-e" and "S-f") and vocabulary (columns "V-e" and "V-f") of $\mathcal{C}^0$ found in $\mathcal{C}^1$ and $\mathcal{C}^2$ is given in Table 3. As one can note, the final corpora obtained through the method presented above preserve most of the information from the base corpus. Especially for the vocabulary, the final corpora cover nearly all the vocabulary of the base corpus. Considering the comparability scores, the comparability of $\mathcal{C}^1$ is 0.912 and the one of $\mathcal{C}^2$ is 0.916. Both of them are more comparable than the base corpus of which the comparability is 0.882.

From these results of the intrinsic evaluation, one can conclude that the strategy developed to improve the corpus quality while preserving most of its information is efficient: The corpora obtained here, $\mathcal{C}^1$ and $\mathcal{C}^2$, are more comparable than the base corpus $\mathcal{C}^0$ and preserve most of its information. We now turn to the problem of extracting bilingual lexicons from these corpora.

5 Bilingual Lexicon Extraction

Following standard practice in bilingual lexicon extraction from comparable corpora, we rely on the approach proposed by Fung and Yee (1998). In this approach, each word w is represented as a

	D-e	D-f	S-e	S-f	V-e	V-f
$\mathcal{C}^1$	0.669	0.698	0.821	0.805	0.937	0.981
$\mathcal{C}^2$	0.785	0.719	0.893	0.807	0.968	0.987

Table 3: Proportion of documents, sentences and vocabulary of $\mathcal{C}^0$ covered by the result corpora

context vector consisting of the weight $a(w^c)$ of each context word w^c, the context being extracted from a window running through the corpus. Once context vectors for English and French words have been constructed, a general bilingual dictionary $\mathcal{D}$ can be used to bridge them by accumulating the contributions from words that are translation of each other. Standard similarity measures, as the cosine or the Jaccard coefficient, can then be applied to compute the similarity between vectors. For example, the cosine leads to:

$$s_c(w_e, w_f) = \frac{\sum_{(w_e^c, w_f^c) \in \mathcal{D}} a(w_e^c) a(w_f^c)}{\|\overrightarrow{w_e}\| \cdot \|\overrightarrow{w_f}\|} \quad (1)$$

5.1 Using Algorithm 1 pseudo-Alignments

The process we have defined in the previous section to improve the quality of a given corpus while preserving its vocabulary makes use of highly comparable document pairs, and thus provides some loose alignments between the two corpora. One can thus try to leverage the above approach to bilingual lexicon extraction by re-weighting $s_c(w_e, w_f)$ by a quantity which is large if w_e and w_f appear in many document pairs with a high comparability score, and small otherwise. In this section, we can not use the alignments in algorithm 1 directly because the alignments in the comparable corpus should not be 1 to 1 and we did not try to find the precise 1 to 1 alignments in algorithm 1.

Let η be the threshold used in algorithm 1 to construct the improved corpus and let $\phi(d_e, d_f)$ be defined as:

$$\phi(d_e, d_f) = \begin{cases} 1 & \text{iff } M(d_e, d_f) \geq \eta \\ 0 & \text{else} \end{cases}$$

Let $\mathcal{H}_e$ (resp. $\mathcal{H}_f$) be the set of documents containing word w_e (resp. w_f). We define the joint probability of w_e and w_f as being proportional to the number of comparable document pairs they belong to, where two documents are comparable if their comparability score is above η, that is:

$$p(w_e, w_f) \propto \sum_{d_e \in \mathcal{H}_e, d_f \in \mathcal{H}_f} \phi(d_e, d_f)$$

The marginal probability $p(w_e)$ can then be written as:

$$\begin{aligned} p(w_e) &\propto \sum_{w_f \in \mathcal{C}_f^v} p(w_e, w_f) \\ &\propto \sum_{d_e \in \mathcal{H}_e} \sum_{d_f \in \mathcal{C}_f^d} |d_f| \cdot \phi(d_e, d_f) \end{aligned}$$

Assuming that all d_f in $\mathcal{C}_f^d$ have roughly the same vocabulary size and all d_e have the same number of comparable counterparts in $\mathcal{C}_f^d$, then the marginal probability can be simplified as: $p(w_e) \propto |\mathcal{H}_e|$. By resorting to the exponential of the point-wise mutual information, one finally obtains the following weight:

$$\begin{aligned} \pi(w_e, w_f) &= \frac{p(w_e, w_f)}{p(w_e) \cdot p(w_f)} \\ &\propto \frac{1}{|\mathcal{H}_e| \cdot |\mathcal{H}_f|} \sum_{d_e \in \mathcal{H}_e, d_f \in \mathcal{H}_f} \phi(d_e, d_f) \end{aligned}$$

which has the desired property: It is large if the two words appear in comparable document pairs more often than chance would predict, and small otherwise. We thus obtain the revised similarity score for w_e and w_f:

$$s_{cr}(w_e, w_f) = s_c(w_e, w_f) \cdot \pi(w_e, w_f) \quad (2)$$

5.2 Validation

In order to measure the performance of the bilingual lexicon extraction method presented above, we divided the original dictionary into 2 parts: 10% of the English words (3,338 words) together with their translations are randomly chosen and used as the evaluation set, the remaining words (30,034 words) being used to compute context vectors and similarity between them. In this study, the weight $a(w^c)$ used in the context vectors (see above) are taken to be the tf-idf score of w^c: $a(w^c)$ = tf-idf(w^c). English words not present in $\mathcal{C}_e^v$ or with no translation in $\mathcal{C}_f^v$ are excluded from the evaluation set. For each English word in the evaluation set, all the French words in $\mathcal{C}_f^v$ are then ranked according to their similarity with the English word (using either equation 1 or 2). To evaluate the quality of the lexicons extracted, we first retain for each English word its N first translations, and then measure the precision of the lists obtained, which amounts in this case to the proportion of lists containing the correct translation (in case of multiple translations, a list is deemed to contain the correct translation as soon as one of the possible translations is present). This evaluation procedure has been used in previous work (e.g. (Gaussier et al., 2004)) and is now standard for the evaluation of lexicons extracted from comparable corpora. In this study, N is set to 20. Furthermore, several studies have shown that it is easier to find the correct translations for frequent words than for infrequent ones (Pekar et al., 2006). To take this fact into account, we distinguished different frequency ranges to assess the validity of our approach for all frequency ranges. Words with frequency less than 100 are defined as low-frequency words (W_L), whereas words with frequency larger than 400 are high-frequency words (W_H), and words with frequency in between are medium-frequency words (W_M).

We then tested the standard method based on the cosine similarity (equation 1) on the corpora $\mathcal{C}^0$, $\mathcal{C}_H$, $\mathcal{C}'_H$, $\mathcal{C}^1$ and $\mathcal{C}^2$. The results obtained are displayed in Table 4, and correspond to columns 2-6. They show that the standard approach performs significantly better on the improved corpora $\mathcal{C}^1$/$\mathcal{C}^2$ than on the base corpus $\mathcal{C}^0$. The overall precision is increased by 5.3% on $\mathcal{C}^1$ (corresponding to a relative increase of 26%) and 9.5% on $\mathcal{C}^2$ (corresponding to a relative increase of 51%), even though the low-frequency words, which dominate the overall precision, account for a higher proportion in $\mathcal{C}^1$ (61.3%) and $\mathcal{C}^2$ (61.3%) than in $\mathcal{C}^0$ (56.2%). For the medium and high frequency words, the precision is increased by over 11% on $\mathcal{C}^1$ and 16% on $\mathcal{C}^2$. As pointed out in other studies, the performance for the low-frequency words is usually bad due to the lack of context information. This explains the relatively small improvement obtained here (only 2.2% on $\mathcal{C}^1$ and 6.7% on $\mathcal{C}^2$). It should also be noticed that the performance of the standard approach is better on $\mathcal{C}^2$ than on $\mathcal{C}^1$, which may be due to the fact that $\mathcal{C}^2$ is slightly larger than $\mathcal{C}^1$ and thus provides more information or to the actual content of these corpora. Lastly, if we consider the results on the corpus $\mathcal{C}_H$ which is produced by only choosing the highly comparable part from $\mathcal{C}^0$, the overall precision is increased by only 1.9%, which might come from the fact that the size of $\mathcal{C}_H$ is less than half the size of $\mathcal{C}^0$. We also notice the better results on $\mathcal{C}_H$ than on $\mathcal{C}'_H$ of the same size which consists of randomly choosing documents from $\mathcal{C}^0$.

The results obtained with the refined approach making use of the comparable document pairs found in the improved corpus (equation 2) are also displayed in Table 4 (columns "$\mathcal{C}^1$ new" and "$\mathcal{C}^2$ new"). From these results, one can see that the overall precision is further improved by 2.0% on $\mathcal{C}^1$ and 2.3% on $\mathcal{C}^2$, compared with the standard approach. For all the low, medium and high-frequency words, the precision has been improved, which demonstrates that the information obtained through the corpus enrichment process contributes to improve the quality of the extracted bilingual lexicons. Compared with the original base corpus $\mathcal{C}^0$, the overall improvement of the precision on both $\mathcal{C}^1$ and $\mathcal{C}^2$ with the refined approach is significant and important (respectively corresponding to a relative improvement of 35% and 62%), which also demonstrates that the efficiency of the refined approach is not confined to a specific external corpus.

6 Discussion

It is in a way useless to deploy bilingual lexicon extraction techniques if translation equivalents are not present in the corpus. This simple fact is at the basis of our approach which consists in constructing comparable corpora close to the original corpus and which are more likely to contain translation equivalents as they have a guaranteed degree of comparability. The pseudo-alignments identified in the construction process are then used to leverage state-of-the-art bilingual lexicon extraction methods. This approach to bilingual lexicon extraction from comparable corpora radically differs, to our knowledge, from previous approaches

	$\mathcal{C}^0$	$\mathcal{C}_H$	$\mathcal{C}'_H$	$\mathcal{C}^1$	$\mathcal{C}^2$	$\mathcal{C}^1$ new	$> \mathcal{C}^1, > \mathcal{C}^0$	$\mathcal{C}^2$ new	$> \mathcal{C}^2, > \mathcal{C}^0$
W_L	0.114	0.144	0.125	0.136	0.181	0.156	2.0%, 4.2%	0.205	2.4%, 9.1%
W_M	0.233	0.313	0.270	0.345	0.401	0.369	2.4%, 3.6%	0.433	3.2%, 20.0%
W_H	0.417	0.456	0.377	0.568	0.633	0.581	1.3%, 16.4%	0.643	1.0%, 22.6%
All	**0.205**	**0.224**	**0.189**	**0.258**	**0.310**	**0.278**	**2.0%, 7.3%**	**0.333**	**2.3%, 12.8%**

Table 4: Precision of the different approaches on different corpora

which are mainly variants of the standard method proposed in (Fung and Yee, 1998) and (Rapp, 1999). For example, the method developed in (Déjean et al., 2002) and (Chiao and Zweigenbaum, 2002) involves a representation of dictionary entries with context vectors onto which new words are mapped. Pekar et al. (2006) smooth the context vectors used in the standard approach in order to better deal with low frequency words. A nice geometric interpretation of these processes is proposed in (Gaussier et al., 2004), which furthermore introduces variants based on Fisher kernels, Canonical Correlation Analysis and a combination of them, leading to an improvement of the F1-score of 2% (from 0.14 to 0.16) when considering the top 20 candidates. In contrast, the approach we have developed yields an improvement of 7% (from 0.13 to 0.20) of the F-1 score on $\mathcal{C}^2$, again considering the top 20 candidates. More important, however, is the fact that the approach we have developed can be used in conjunction with any existing bilingual extraction method, as the strategies for improving the corpus quality and the re-weighting formula (equation 2) are general. We will assess in the future whether substantial gains are also attained with other methods.

Some studies have tried to extract subparts of comparable corpora to complement existing parallel corpora. Munteanu (2004) thus developed a maximum entropy classifier aiming at extracting those sentence pairs which can be deemed parallel. The step for choosing similar document pairs in this work resembles some of our steps. However their work focuses on high quality and specific documents pairs, as opposed to the entire corpus of guaranteed quality we want to build. In this latter case, the cross-interaction between documents impacts the overall comparability score, and new methods, as the one we have introduced, need to be proposed. Similarly, Munteanu and Marcu (2006) propose a method to extract sub-sentential fragments from non-parallel corpora. Again, the targeted elements are very specific (parallel sentences or sub-sentences) and limited, and the focus is put on a few sentences which can be considered parallel. As already mentioned, we rather focus here on building a new corpus which preserves most of the information in the original corpus. The construction process we have presented is theoretically justified and allows one to preserve ca. 95% of the original vocabulary.

7 Conclusion

We have first introduced in this paper a comparability measure based on the expectation of finding translation word pairs in the corpus. We have then designed a strategy to construct an improved comparable corpus by (a) extracting a subpart of the original corpus with a guaranteed comparability level, and (b) by completing the remaining subpart with external resources, in our case other existing bilingual corpora. We have then shown how the information obtained during the construction process could be used to improve state-of-the-art bilingual lexicon extraction methods. We have furthermore assessed the various steps of our approach and shown: (a) that the comparability measure we introduced captures variations in the degree of comparability between corpora, (b) that the construction process we introduced leads to an improved corpus preserving most of the original vocabulary, and (c) that the use of pseudo-alignments through simple re-weighting yields bilingual lexicons of higher quality.

Acknowledgements

This work was supported by the French National Research Agency grant ANR-08-CORD-009.

References

Ballesteros, Lisa and W. Bruce Croft. 1997. Phrasal translation and query expansion techniques for cross-language information retrieval. In *Proceedings of the 20th ACM SIGIR*, pages 84–91, Philadelphia, Pennsylvania, USA.

Chen, Stanley F. 1993. Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st Annual Conference of the Association for Computational Linguistics*, pages 9–16, Columbus, Ohio, USA.

Chiao, Yun-Chuang and Pierre Zweigenbaum. 2002. Looking for candidate translational equivalents in specialized, comparable corpora. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan.

Déjean, Hervé, Eric Gaussier, and Fatia Sadat. 2002. An approach based on multilingual thesauri and model combination for bilingual lexicon extraction. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7, Taipei, Taiwan.

Fung, Pascale and Kathleen McKeown. 1997. Finding terminology translations from non-parallel corpora. In *Proceedings of the 5th Annual Workshop on Very Large Corpora*, pages 192–202, Hong Kong.

Fung, Pascale and Lo Yuen Yee. 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 17th international conference on Computational linguistics*, pages 414–420, Montreal, Quebec, Canada.

Gaussier, E., J.-M. Renders, I. Matveeva, C. Goutte, and H. Déjean. 2004. A geometric view on bilingual lexicon extraction from comparable corpora. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 526–533, Barcelona, Spain.

Kay, Martin and Martin Röscheisen. 1993. Text-translation alignment. *Computational Linguistics*, 19(1):121–142.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of MT Summit 2005*.

Melamed, I. Dan. 1997a. A portable algorithm for mapping bitext correspondence. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 305–312, Madrid, Spain.

Melamed, I. Dan. 1997b. A word-to-word model of translational equivalence. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and the 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 490–497, Madrid, Spain.

Morin, Emmanuel, Béatrice Daille, Koichi Takeuchi, and Kyo Kageura. 2007. Bilingual terminology mining - using brain, not brawn comparable corpora. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 664–671, Prague, Czech Republic.

Munteanu, Dragos Stefan and Daniel Marcu. 2006. Extracting parallel sub-sentential fragments from non-parallel corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 81–88, Sydney, Australia.

Munteanu, Dragos Stefan, Alexander Fraser, and Daniel Marcu. 2004. Improved machine translation performance via parallel sentence extraction from comparable corpora. In *Proceedings of the HLT-NAACL 2004*, pages 265–272, Boston, MA., USA.

Och, Franz Josef and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Pekar, Viktor, Ruslan Mitkov, Dimitar Blagoev, and Andrea Mulloni. 2006. Finding translations for low-frequency words in comparable corpora. *Machine Translation*, 20(4):247–266.

Rapp, Reinhard. 1999. Automatic identification of word translations from unrelated English and German corpora. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 519–526, College Park, Maryland, USA.

Robitaille, Xavier, Yasuhiro Sasaki, Masatsugu Tonoike, Satoshi Sato, and Takehito Utsuro. 2006. Compiling French-Japanese terminologies from the web. In *Proceedings of the 11st Conference of the European Chapter of the Association for Computational Linguistics*, pages 225–232, Trento, Italy.

Yu, Kun and Junichi Tsujii. 2009. Extracting bilingual dictionary from comparable corpora with dependency heterogeneity. In *Proceedings of HLT-NAACL 2009*, pages 121–124, Boulder, Colorado, USA.

Structure-Aware Review Mining and Summarization

Fangtao Li[1], Chao Han[1], Minlie Huang[1], Xiaoyan Zhu[1], Ying-Ju Xia[2], Shu Zhang[2] and Hao Yu[2]

[1]State Key Laboratory of Intelligent Technology and Systems，
[1]Tsinghua National Laboratory for Information Science and Technology，
[1]Department of Computer Science and Technology, Tsinghua University
[2]Fujitsu Research and Development Center

fangtao06@gmail.com; zxy_dcs@tsinghua.edu.cn

Abstract

In this paper, we focus on object feature [1] based review summarization. Different from most of previous work with linguistic rules or statistical methods, we formulate the review mining task as a joint structure tagging problem. We propose a new machine learning framework based on Conditional Random Fields (CRFs). It can employ rich features to jointly extract positive opinions, negative opinions and object features for review sentences. The linguistic structure can be naturally integrated into model representation. Besides linear-chain structure, we also investigate conjunction structure and syntactic tree structure in this framework. Through extensive experiments on movie review and product review data sets, we show that structure-aware models outperform many state-of-the-art approaches to review mining.

1 Introduction

With the rapid expansion of e-commerce, people are more likely to express their opinions and hands-on experiences on products or services they have purchased. These reviews are important for both business organizations and personal costumers. Companies can decide on their strategies for marketing and products improvement. Customers can make a better decision when purchasing products or services. Unfortunately, reading through all customer reviews is difficult, especially for popular items, the number of reviews can be up to hundreds or even thousands. Therefore, it is necessary to provide coherent and concise summaries for these reviews.

Gone With The Wind:
Movie:
Positive: great, good, amazing, … , breathtaking
Negative: bad, boring, waste time, … , mistake
Actor:
Positive: charming , brilliant , great, … , smart
Negative: poor, fail, dirty, … , lame
Music:
Positive: great, beautiful, very good, … , top
Negative: annoying, noise, too long, … , unnecessary
… …

Figure 1. Feature based Review Summarization

Inspired by previous work (Hu and Liu, 2004; Jin and Ho, 2009), we aim to provide object feature based review summarization. Figure 1 shows a summary example for movie "Gone with the wind". The object (movie) features, such as "movie", "actor", with their corresponding positive opinions and negative opinions, are listed in a structured way. The opinions are ranked by their frequencies. This provides a concise view for reviews. To accomplish this goal, we need to do three tasks: 1), extract all the object features and opinions; 2), determine the sentiment polarities for opinions; 3), for each object feature, determine the relevant opinions, i.e. object feature-opinion pairs.

For the first two tasks, most previous studies employ linguistic rules or statistical methods (Hu and Liu, 2004; Popescu and Etzioni 2005). They mainly use unsupervised learning methods, which lack an effective way to address infrequent object features and opinions. They are also hard to incorporate rich overlapping features.

[1] Note that there are two meanings for word "feature". We use "object feature" to represent the target entity, which the opinion expressed on, and use "feature" as the input for machine learning methods.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 653–661,
Beijing, August 2010

Actually, there are many useful features, which have not been fully exploited for review mining. Meanwhile, most of previous methods extract object features, opinions, and determine the polarities for opinions separately. In fact, the object features, positive opinions and negative opinions correlate with each other.

In this paper, we formulate the first two tasks, i.e. object feature, opinion extraction and opinion polarity detection, as a joint structure tagging problem, and propose a new machine learning framework based on Conditional Random Fields (CRFs). For each sentence in reviews, we employ CRFs to jointly extract object features, positive opinions and negative opinions, which appear in the review sentence. This framework can naturally encode the linguistic structure. Besides the neighbor context with linear-chain CRFs, we propose to use Skip-chain CRFs and Tree CRFs to utilize the conjunction structure and syntactic tree structure. We also propose a new unified model, Skip-Tree CRFs to integrate these structures. Here, "structure-aware" refers to the output structure, which model the relationship among output labels. This is significantly different from the previous input structure methods, which consider the linguistic structure as heuristic rules (Ding and Liu, 2007) or input features for classification (Wilson et al. 2009). Our proposed framework has the following advantages: First, it can employ rich features for review mining. We will analyze the effect of features for review mining in this framework. Second, the framework can utilize the relationship among object features, positive opinions and negative opinions. It jointly extracts these three types of expressions in a unified way. Third, the linguistic structure information can be naturally integrated into model representation, which provides more semantic dependency for output labels. Through extensive experiments on movie review and product review, we show our proposed framework is effective for review mining.

The rest of this paper is organized as follows: In Section 2, we review related work. We describe our structure aware review mining methods in Section 3. Section 4 demonstrates the process of summary generation. In Section 5, we present and discuss the experiment results. Section 6 is the conclusion and future work.

2 Related Work

Object feature based review summary has been studied in several papers. Zhuang et al. (2006) summarized movie reviews by extracting object feature keywords and opinion keywords. Object feature-opinion pairs were identified by using a dependency grammar graph. However, it used a manually annotated list of keywords to recognize movie features and opinions, and thus the system capability is limited. Hu and Liu (2004) proposed a statistical approach to capture object features using association rules. They only considered adjective as opinions, and the polarities of opinions are recognized with WordNet expansion to manually selected opinion seeds. Popescu and Etzioni (2005) proposed a relaxation labeling approach to utilize linguistic rules for opinion polarity detection. However, most of these studies focus on unsupervised methods, which are hard to integrate various features. Some studies (Breck et al. 2007; Wilson et al, 2009; Kobayashi et al. 2007) have used classification based methods to integrate various features. But these methods separately extract object features and opinions, which ignore the correlation among output labels, i.e. object features and opinions. Qiu et al. (2009) exploit the relations of opinions and object features by adding some linguistic rules. However, they didn't care the opinion polarity. Our framework can not only employ various features, but also exploit the correlations among the three types of expressions, i.e. object features, positive opinions, and negative opinions, in a unified framework. Recently, Jin and Ho (2009) propose to use Lexicalized HMM for review mining. Lexicalized HMM is a variant of HMM. It is a generative model, which is hard to integrate rich, overlapping features. It may encounter sparse data problem, especially when simultaneously integrating multiple features. Our framework is based on Conditional Random Fields (CRFs). CRFs is a discriminative model, which can easily integrate various features.

These are some studies on opinion mining with Conditional Random Fields. For example, with CRFs, Zhao et al (2008) and McDonald et al. (2007) performed sentiment classification in sentence and document level; Breck et al (2007) identified opinion expressions from newswire documents; Choi et al. (2005) determined opinion holders to opinions also from newswire data. None of previous work focuses on jointly extracting object features, positive opinions and negative opinions simultaneously from review data. More importantly, we also show how to encode the linguistic structure, such as conjunction structure and syntactic tree structure, into model representation in our framework. This is significantly different from most of previous studies, which consider the structure information as heuristic rules (Hu and Liu, 2004) or input features (Wilson et al. 2009).

Recently, there are some studies on joint sentiment/topic extraction (Mei et al. 2007; Titov and McDonald, 2008; Snyder and Barzilay, 2007). These methods represent reviews as several coarse-grained topics, which can be considered as clusters of object features. They are hard to indentify the low-frequency object features and opinions. While in this paper, we will extract all the present object features and corresponding opinions with their polarities. Besides, the joint sentiment/topic methods are mainly based on review document for topic extraction. In our framework, we focus on sentence-level review extraction.

3 Structure Aware Review Mining

3.1 Problem Definition

To produce review summaries, we need to first finish two tasks: identifying object features, opinions, and determining the polarities for opinions. In this paper, we formulate these two tasks as a joint structure tagging problem. We first describe some related definitions:

Definition (Object Feature): is defined as whole target expression that the subjective expressions have been commented on. Object features can be products, services or their elements and properties, such as "character", "movie", "director" for movie review, and "battery", "battery life", "memory card" for product review.

Definition (Review Opinion): is defined as the whole subjective expression on object features. For example, in sentence "The camera is easy to use", "easy to use" is a review opinion. "opinion" is used for short.

Definition (Opinion Polarity): is defined as the sentiment category for review opinion. In this paper, we consider two types of polarities: positive opinion and negative opinion. For example, "easy to use" belongs to positive opinion.

For our review mining task, we need to represent three types of expressions: object features, positive opinions, and negative opinions. These expressions may be words, or whole phrases. We use BIO encoding for tag representation, where the non-opinion and neutral opinion words are represented as "O". With Negation (N), which is only one word, such as "not", "don't", as an independent tag, there are totally 8 tags, as shown in Table 1. The following is an example to denote the tags:

The/*O* camera/*FB* comes/*O* with/*O* a/*O* pitiful/*CB* 32mb/*FB* compact/*FI* flash/*FI* card/*FI* ./*O*

FB	Feature Beginning	CB	Negative Beginning
FI	Feature Inside	CI	Negative Inside
PB	Positive Beginning	N	Negation Word
PI	Positive Inside	O	Other

Table 1. Basic Tag Set for Review Mining

3.2 Structure Aware Model

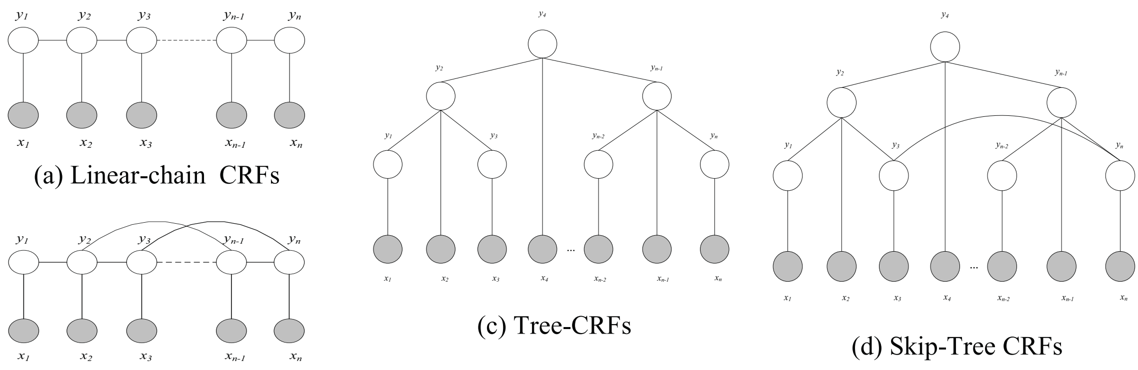
In this section, we describe how to encode different linguistic structure into model representation based on our CRFs framework.

3.2.1 Using Linear CRFs.

For each sentence in a review, our task is to extract all the object features, positive opinions and negative opinions. This task can be modeled as a classification problem. Traditional classification tools, e.g. Maximum Entropy model (Berger et al, 1996), can be employed, where each word or phrase will be treated as an instance. However, they independently consider each word or phrase, and ignore the dependency relationship among them.

Actually, the context information plays an important role for review mining. For example, given two continuous words with same part of speech, if the previous word is a positive opinion, the next word is more likely a positive opinion. Another example is that if the previous word is an adjective, and it is an opinion, the next noun word is more likely an object feature.

To this end, we formulate the review mining task as a joint structure tagging problem, and propose a general framework based on Conditional Random Fields (CRFs) (Lafferty et al., 2001) which are able to model the dependencies



(a) Linear-chain CRFs

(b) Skip-chain CRFs

(c) Tree-CRFs

(d) Skip-Tree CRFs

Figure 2 CRFs models

between nodes. (See Section 3.2.5 for more about CRFs)

In this section, we propose to use linear-chain CRFs to model the sequential dependencies between continuous words, as discussed above. It views each word in the sentence as a node, and adjacent nodes are connected by an edge. The graphical representation is shown in Figure 2(a). Linear CRFs can make use of dependency relationship among adjacent words.

3.2.2 Leveraging Conjunction Structure

We observe that the conjunctions play important roles on review mining: If the words or phrases are connected by conjunction "and", they mostly belong to the same opinion polarity. If the words or phrases are connected by conjunction "but", they mostly belong to different opinion polarity, as reported in (Hatzivassiloglou and McKeown, 1997; Ding and Liu, 2007). For example, "This phone has a very cool and useful feature – the speakerphone", if we only detect "cool", it is hard to determine its opinion polarity. But if we see "cool" is connected with "useful" by conjunction "and", we can easily acquire the polarity of "cool" as positive. This conjunction structure not only helps to determine the opinions, but also helps to recognize object features. For example, "I like the special effects and music in this movie", with word "music" and conjunction "and", we can easily detect that "special effects" as an object feature.

To model the long distance dependency with conjunctions, we use Skip-chain CRFs model to detect object features and opinions. The graphical representation of a Skip-chain CRFs, given in Figure 2(b), consists of two types of edges: linear-edge (y_{t-1} to y_t) and skip-edge (y_i to y_j). The linear-edge is described as linear CRFs. The skip-edge is imported as follows:

We first identify the conjunctions in the review sentence, with a collected conjunction set, including "and", "but", "or", "however", "although" etc. For each conjunction, we extract its connected two text sequences. The nearest two words with same part of speech from the two text sequences are connected with the skip-edge. Here, we just consider the noun, adjective, and adverb. For example, in "good pictures and beautiful music", there are two skip-edges: one connects two adjective words "good" and "beautiful"; the other connects two nouns "pictures" and "music". We also employ the general sentiment lexicons, SentiWordNet (Esuli and Sebastiani, 2006), to connect opinions. Two nearest opinion words, detected by sentiment lexicon, from two sequences, will also be connected by skip-edge. If the nearest distance exceeds the threshold, this skip edge will be discarded. Here, we consider the threshold as nine.

Skip-chain CRFs improve the performance of review mining, because it naturally encodes the conjunction structure into model representation with skip-edges.

3.2.3 Leveraging Syntactic Tree Structure

Besides the conjunction structure, the syntactic tree structure also helps for review mining. The tree denotes the syntactic relationship among words. In a syntactic dependency representation, each node is a surface word. For example, the corresponding dependency tree (Klein and Manning, 2003) for the sentence, "I really like this long movie", is shown in Figure 3.

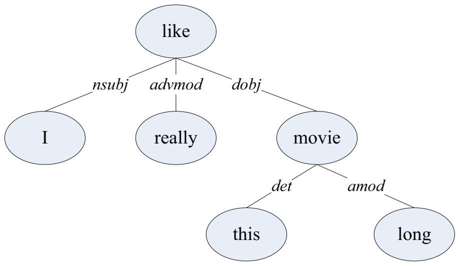


Figure 3. Syntactic Dependency Tree Representation

In linear-chain structure and skip-chain structure, "like" and "movie" have no direct edge, but in syntactic tree, "movie" is directly connected with "like", and their relationship "dobj" is also included, which shows "movie" is an objective of "like". It can provide deeper syntactic dependencies for object features, positive opinions and negative opinions. Therefore, it is important to consider the syntactic structure in the review mining task.

In this section, we propose to use Tree CRFs to model the syntactic tree structure for review mining. The representation of a Tree CRFs is shown in Figure 2(c). The syntactic tree structure is encoded into our model representation. Each node is corresponding to a word in the dependency tree. The edge is corresponding to dependency tree edge. Tree CRFs can make use of dependency relationship in syntactic tree structure to boost the performance.

3.2.4 Integrating Conjunction Structure and Syntactic Tree Structure

Conjunction structure provides the semantic relations correlated with conjunctions. Syntactic tree structure provides dependency relation in the syntactic tree. They represent different semantic dependencies. It is interesting to consider these two dependencies in a unified model. We propose Skip-Tree CRFs, to combine these two structure information. The graphical representation of a Skip-Tree CRFs, given in Figure 2(d), consists of two types of edges: tree edges and conjunction skip-edges. We hope to simultaneously model the dependency in conjunction structure and syntactic tree structure.

We also notice that there is a relationship "conj" in syntactic dependency tree. However, we find that it only connects two head words for a few coordinating conjunction, such as "and", "or", "but". Our designed conjunction skip-edge provides more information for joint structure tagging. We analyze more conjunctions to connect not only two head words, but also the words with same part of speech. We also connect the words with sentiment lexicon. We will show that the skip-tree CRFs, which combine the two structures, is effective in the experiment section.

3.2.5 Conditional Random Fields

A CRFs is an undirected graphical model G of the conditional distribution $P(Y|X)$. Y are the random variables over the labels of the nodes that are globally conditioned on *X*, which are the random variables of the observations. The conditional probability is defined as:

$$\mathrm{P}(Y\,|X) = \frac{1}{Z(X)}\, exp\left(\sum_{e\in E,i} \gamma_i t_i(e, Y|e, X) + \sum_{v\in V,i} \mu_i s_i(v, Y|v, X)\right)$$

where *Z(x)* is the normalization factor, s_i is the state function on node, t_i is the transition functions on edge, and r_i and μ_i are parameters to estimate (Sutton and McCallum, 2006).

Inference and Parameter Estimation. For Linear CRFs, dynamic programming is used to compute the maximum a posteriori (MAP) of *Y* given *X*. For more complicated graphs with cycles, we employ Tree Re-Parameterization (TRP) algorithm (Wainwright et al. 2001) for approximate inference.

Given the training Data $D = \{x^{(i)}, y^{(i)}\}_{i=1}^{n}$, the parameter estimation is to determine the parameters based on maximizing the log-likelihood $L_\theta = \sum_{i=1}^{n} log\, p(y^{(i)}|x^{(i)})$. In Linear CRFs model, dynamic programming and L-BFGS algorithm can be used to optimize objective function L_θ, while for complicated CRFs, TRP is used instead to calculate the marginal probability.

3.3 Feature Space

In this section, we describe the features used in the learning methods. All the features are listed in Figure 4. Word features include the word's token, lemma, and part of speech. The adjacent words' information is considered. We detect whether the negation words appear in the previous four words as a binary feature. We also detect whether this word is the superlative form, such as "best", and comparative form, such as "better", as binary features. Two types of dictionaries are employed. We use WordNet to acquire the synonyms and antonyms for each word. SentiWordNet (Esuli and Sebastiani, 2006) is used to acquire the prior polarity for each word. We use the words with positive or negative score

Word Feature:
- Word token
- Word lemma
- Word part of speech
- Previous word token, lemma, part of speech
- Next word token, lemma, part of speech
- Negation word appears in previous 4 words
- Is superlative degree
- Is comparative degree

Dictionary Feature
- WordNet Synonym
- WordNet Antonym
- SentiWordNet Prior Polarity

Sentence Feature
- Num of positive words in SentiWordNet
- Num of negative words in SentiWordNet
- Num of Negation word

Syntactic Features:
- Parent word
- Parent SentiWordnet Prior Polarity
- In subject
- In copular
- In object

Edge Feature
- Conjunction word
- Syntactic relationship

Figure 4. Features for learning Methods

above a threshold (0.6). Sentence Feature provides sentence level information. It includes the count of positive words and negative words, which are detected by SentiWordNet. We also incorporate the count of negation words as a feature. There are some syntactic features from dependency tree. Parent word and its polarity are considered. We also detect if the word is subject, object or copular. For edge features, the conjunction words are incorporated as corresponding skip-edge features. The syntactic relationship is considered as a feature for corresponding tree-edge. For classification and linear CRFs models, we just add this edge features as general features.

4 Review Summary Generation

After extracting the object features and opinions, we need to extract the relevant opinions for each feature. In this paper, we identify the nearest opinion word/phrase for each object feature as object feature-opinion pair, which is widely used in previous work (Hu and Liu, 2004; Jin and Ho, 2009). The review summary is generated as a list of structured object feature-opinion pairs, as shown in Figure 1.

5 Experiment

5.1 Experiment setup

Data Set: For our structure tagging task, we need to know the labels for all the words in reviews. In this paper, we manually annotate two types of these review data sets. One is movie review, which contains five movies with totally 500 reviews. The other is product review, which contains four products with totally 601 reviews. We need to label all object features, positive opinions, negative opinions, and the object feature-opinion pairs for all sentences. Each sentence is labeled by two annotators. The conflict is checked by the third person. Finally, we acquire 2207 sentences for movie review and 2533 sentences for product review. For each type, including movie and product, the data set is divided into five parts. We select four parts as training data, and the fifth part as testing data.

Evaluation Metric:
Precision, Recall and F measure are used to test our results, as Jin and Ho (2009).

5.2 Baselines

First word	Second Word	Third Word
JJ	NN or NNS	Anything
RB, RBR or RBS	JJ	NN or NNS
JJ	JJ	NN or NNS
NN or NNS	JJ	Not NN or NNS

Table 2. Rules in rule based method

Rule based Method:
The rule based method is used in Jin and Ho (2009), which is motivated by (Hu and Liu, 2004; Turney, 2002). The employed rules are shown in Table 2. The matching adjective is identified as opinion, and matching nouns are extracted as object features. To determine the polarities of the opinions, 25 positive adjectives and 25 negative adjectives are used as seeds, and then expanded by searching synonyms and antonyms in WordNet. The polarity of a word is detected by checking the collected lists.

Lexicon based Method:
The object features and opinions extraction is same as rule based method. The general sentiment lexicon SentiWordNet is employed to detect the polarity for each word.

Lexicalized HMM:
The object features and opinions are identified by Lexicalized HMM (L-HMM), as Jin and Ho (2009). L-HMM is a variant of HMM. It has two observations. The current tag is not only related

	Methods	Object Features			Positive Opinions			Negative Opinions			Overall		
		P(%)	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*
Movie Review	Rule	41.2	32.3	36.2	**82.9**	31.1	45.3	23.5	13.7	17.3	49.2	25.7	33.8
	Lexicon	41.2	32.3	36.2	64.0	38.1	47.8	19.6	6.8	10.2	41.6	25.8	31.8
	L-HMM	**88.0**	52.6	65.9	82.1	49.6	61.9	65.9	**41.1**	**50.6**	78.7	47.8	59.5
	MaxEnt	83.4	75.1	79.1	82.2	**65.0**	**72.6**	74.1	29.5	42.2	**79.9**	56.5	66.2
	Linear CRFs	81.8	**78.4**	**80.1**	79.1	63.9	70.7	**75.8**	32.2	45.2	79.0	**58.2**	**67.0**
Product Review	Rule	53.5	35.6	42.8	74.4	22.5	34.6	17.1	8.9	11.7	48.3	22.3	30.6
	Lexicon	53.5	35.6	42.8	48.9	29.7	40.0	14.7	3.7	5.9	39.1	23.0	29.0
	L-HMM	83.9	48.7	61.6	**90.3**	56.8	69.8	47.2	25.2	32.9	73.8	43.6	54.8
	MaxEnt	83.4	55.1	66.4	82.2	65.0	72.6	64.1	30.0	40.4	76.6	49.9	60.4
	Linear CRFs	**91.1**	**56.3**	**69.6**	88.7	**70.4**	**78.5**	**67.7**	**32.6**	**44.0**	**82.5**	**53.1**	**64.6**

Table 3. Comparison Results with Baselines

(the learning methods only employ word token and part of speech as features).

	Methods	Object Features			Positive Opinions			Negative Opinions			Overall		
		P(%)	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*
Movie Review	MaxEnt	82.8	76.6	79.6	80.3	67.8	73.5	**82.8**	36.3	50.5	81.9	60.2	69.4
	Linear CRFs	83.5	75.4	79.2	77.8	71.4	74.5	70.9	53.4	60.9	77.4	66.8	71.7
	Skip CRFs	83.9	78.7	81.2	81.8	73.4	77.4	75.2	62.3	68.2	80.3	71.5	75.7
	Tree CRFs	84.1	79.0	81.5	**82.7**	75.4	78.9	76.7	61.0	67.9	81.2	72.2	76.2
	SkipTreeCRFs	**85.5**	**82.0**	**83.7**	82.3	**80.0**	**81.1**	80.2	**66.4**	**72.7**	**82.6**	**76.2**	**79.3**
Product Review	MaxEnt	80.0	70.8	75.1	85.6	65.7	74.3	65.1	37.8	47.8	76.9	58.1	66.2
	Linear CRFs	84.0	72.9	78.1	86.7	72.0	78.6	60.4	49.6	54.5	77.0	64.8	70.4
	Skip CRFs	84.8	73.5	78.7	87.8	74.5	80.6	73.1	50.4	59.6	81.2	66.1	73.2
	Tree CRFs	83.0	72.7	77.5	86.6	73.4	79.4	64.3	54.8	59.2	78.0	67.0	72.1
	SkipTreeCRFs	**87.1**	**74.1**	**80.1**	**91.8**	**76.7**	**83.6**	**81.1**	**57.0**	**67.0**	**86.6**	**69.3**	**77.0**

Table 4. Comparative experiments with all features

with the previous tag, but also correlates with previous observations. They use word token and part of speech as two features.

Classification based Method:

We also formulate the review mining as a classification task. Each word is considered as an instance. Maximum Entropy (MaxEnt) is used in this paper.

5.3 Experiment results

Since Lexicalized HMM employ word token and part of speech as features (Jin and Ho, 2009), we first conduct comparative experiments with these two features for learning methods. Table 3 shows the results. The rule based method is a little better than lexicon based method. SentiWordNet is designed for general opinion mining, which may be not suitable for domain specific review mining task. For rule based method, the seeds are selected in the review domain, which is more suitable for domain specific task. However, both methods achieve low performance. This because that they only employ simple linguistic rules to extract object features and opinions, which is not effective for infrequent cases and phrase cases. Lexicalized HMM is an extension of HMM. It uses word token and part of speech as two observations. The current tag is not only related with the previous tag, but also correlates with previous two observations. Lexicalized HMM can employ dependency relationship among adjacent words. However, it doesn't achieve the expected result. This is because that Lexicalized HMM is a generative model, which is hard to incorporate rich overlapping features. Even Lexicalized HMM uses linear interpolation smoothing technique. The data sparsity problem seriously hurt the performance. There are many sentences with zero probability. MaxEnt classifier is a discrimitive model, which can incorporate various features. However, it independently classifies each word, and ignores the dependency among successive words. The linear CRFs model achieves best performances for movie review, and product review in overall F-score. This is because that, in our joint structure tagging framework, linear CRFs can employs the global structure to make use of the adjacent dependency relation, and easily incorporate various features to boost the performance.

We also conduct the comparative experiments with all features. From Table 4, we can see that linear CRFs, which consider the chain structure,

	Object Features			Positive Opinions			Negative Opinions			Overall		
	P(%)	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*	*P(%)*	*R(%)*	*F(%)*
Basic	83.8	79.2	81.4	79.5	71.0	75.0	76.1	37.0	49.8	79.8	62.4	70.0
Basic +Word Feature	84.0	81.4	82.7	79.2	75.6	77.4	78.9	48.6	60.2	80.7	68.6	74.1
Basic +Dictionary	80.5	76.6	78.5	**82.7**	76.3	79.4	76.5	60.3	67.4	80.0	71.0	75.2
Basic +Sentence	82.5	75.6	78.9	80.4	75.4	77.8	**84.0**	46.7	60.0	82.3	65.9	73.2
Basic +Syntactic	84.5	70.8	77.0	79.6	73.9	76.7	79.5	47.9	59.8	81.2	64.2	71.7
Basic + Edge	84.1	80.1	82.1	79.5	75.4	77.4	82.4	47.9	60.6	82.0	67.8	74.2
All Features	**85.5**	**82.0**	**83.7**	82.3	**80.0**	**81.1**	80.2	**66.4**	**72.7**	**82.6**	**76.2**	**79.3**

Table 5. Feature Evaluations with Skip Tree CRFs (movie)

still achieve better results than MaxEnt classifier method. Skip-chain CRFs model the conjunction structure in the sentence. We can see that the Skip-chain CRFs achieve better results than linear CRFs. This shows that conjunction structure is really important for review mining. For example "although this camera takes great pictures, it is extremely fragile.", "fragile" is not correctly classified by MaxEnt and Linear CRFs. But the Skip-chain CRFs can correctly classify "fragile" as negative opinion, with conjunction "although", and the skip edge between "great" and "fragile". Tree CRFs encode the syntactic tree structure into model representation. Compared with linear-CRFs, the performances are improved for most of expression identification tasks, except for a little decline for product object feature, which may be because that the tags "FB" and "FI" are out of order when transferring to tree structure. These are no significant difference between Skip-Chain CRFs and Tree CRFs. Conjunction structure and syntactic structure represent the semantic dependency from different views. When integrating these two types of dependencies, the Skip-Tree CRFs achieve better overall results than both Skip-Chain CRFs and Tree CRFs.

Table 5 shows the movie review result for Skip Tree model for different types of features. The basic feature only employs word token as feature set. Other features are defined as shown in Figure 4. By adding different features, we find that they all achieve overall improvements than basic feature. The dictionary features are the most important features, especially for positive opinion and negative opinion identification, which shows the importance of prior word's sentiment. Word features also play important roles: Part of speech is reported useful in several papers (such as Jin and Ho, 2009); the superlative and comparative forms are good indicators for opinion words. Syntactic features acquire limited improvement in this experiment. They may overlap with CRF based structure model. We also find that sentence level features contribute to the review mining task. Edge feature is also important. It makes the skip edge and tree edge with the semantic representation. When combing all the features, the result is significantly improved compared with any single feature set, which shows that it is crucial to integrate various features for review mining.

A review summary example, generated by our methods, is shown in Figure 1.

6 Conclusion

In this paper, we formulate the review mining task as a joint structure tagging problem. A new framework based on Conditional Random Fields is proposed. The framework can employ rich features to simultaneously extract object features, positive opinions and negative opinions. With this framework, we investigate the chain structure, conjunction structure and syntactic tree structure for review mining. A new unified model, called skip tree CRFs, is proposed for review mining. Through extensive experiments, we show that our proposed framework is effective. It outperforms many state-of-the-art methods.

In future work, we will improve the object feature-opinion pair detection with other learning methods. We also want to cluster the related object features to provide more concise review summary.

Acknowledgement

This work was partly supported by Chinese NSF grant No.60973104 and No. 60803075, a grand from Fujitsu Research Center, and a grant from the International Development Research Center, Ottawa, Canada. We also thank the anonymous reviewers, Qiang Yang, Lei Zhang and Qiu Long for their valuable comments.

References

A. Berger and Vincent Della Pietra and Stephen A. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics.

E. Breck, Y. Choi, and C. Cardie. 2007. Identifying expressions of opinion in context. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI).

Y. Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005. Identifying Sources of Opinions with Conditional Random Fields and Extraction Patterns. In Proceedings of HLT-EMNLP.

X. Ding and Bing Liu. 2007. The Utility of Linguistic Rules in Opinion Mining. In Proceedings of SIGIR.

A. Esuli and Fabrizio Sebastiani. 2006. SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining. In Proceedings of LREC.

V. Hatzivassiloglou and K. McKeown. 1997. Predicting the semantic orientation of adjectives. Proceedings of the Joint ACL/EACL Conference.

M. Hu and B. Liu. 2004. Mining and Summarizing Customer Reviews. Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'04).

W. Jin, H.H. Ho. 2009. A novel lexicalized HMM-based learning framework for web opinion mining. Proceedings of the 26th Annual International Conference on Machine Learning (ICML 2009).

J. Lafferty, A. McCallum, F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: Proc. 18th International Conf. on Machine Learning (ICML).

D. Klein and Christopher D. Manning. 2003. Fast Exact Inference with a Factored Model for Natural Language Parsing. In Advances in Neural Information Processing Systems 15 (NIPS 2002),

N. Kobayashi, K. Inui, and Y. Matsumoto. 2007. Opinion Mining from Web documents: Extraction and Structurization. Journal of Japanese society for artificial intelligence.

R. McDonald, K. Hannan, T. Neylon, M. Wells, and J. Reynar. 2007. Structured models for fine-to-coarse sentiment analysis. Proceedings of the Association for Computational Linguistics (ACL).

Q. Mei, X. Ling, M. Wondra, H. Su, and C. Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In Proceedings of the 16th international conference on World Wide Web.

A. Popescu and O. Etzioni. 2005. Extracting Product Features and Opinions from Reviews. Proceedings of 2005 Conference on Empirical Methods in Natural Language Processing (EMNLP'05), 339-346.

G. Qiu, B. Liu, J. Bu and C. Chen. 2009. Expanding Domain Sentiment Lexicon through Double Propagation, International Joint Conference on Artificial Intelligence (IJCAI-09).

B. Snyder and R. Barzilay. 2007. Multiple Aspect Ranking using the Good Grief Algorithm", In Proc. of NAACL

C. Sutton, A. McCallum. 2006. An Introduction to Conditional Random Fields for Relational Learning. In "Introduction to Statistical Relational Learning". Edited by Lise Getoor and Ben Taskar. MIT Press.

I. Titov and R. McDonald. 2008. A Joint Model of Text and Aspect Ratings for Sentiment Summarization.In Proceeding of the Association for Computational Linguistics (ACL).

P. D. Turney. 2002. Thumbs up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. Proceedings of Association for Computational Linguistics (ACL'02).

M. Wainwright, T. Jaakkola, and A. Willsky. 2001. Tree-based reparameterization for approximate estimation on graphs with cycles. In Proceedings of Advances in Neural Information Processing Systems (NIPS'2001). pp. 1001-1008.

T. Wilson, Janyce Wiebe, and Paul Hoffmann 2009. Recognizing Contextual Polarity: an exploration of features for phrase-level sentiment analysis. Computational Linguistics 35(3).

J. Zhao, Kang Liu, Gen Wang. 2008. Adding Redundant Features for CRFs-based Sentence Sentiment Classification. In Proceedings of EMNLP.

L. Zhuang, Feng Jing and Xiaoyan Zhu. 2006. Movie Review Mining and Summarization. In Proceedings of CIKM.

Adaptive Development Data Selection for Log-linear Model in Statistical Machine Translation

Mu Li
Microsoft Research Asia
muli@microsoft.com

Yinggong Zhao*
Nanjing University
zhaoyg@nlp.nju.edu.cn

Dongdong Zhang
Microsoft Research Asia
dozhang@microsoft.com

Ming Zhou
Microsoft Research Asia
mingzhou@microsoft.com

Abstract

This paper addresses the problem of dynamic model parameter selection for log-linear model based statistical machine translation (SMT) systems. In this work, we propose a principled method for this task by transforming it to a test data dependent development set selection problem. We present two algorithms for automatic development set construction, and evaluated our method on several NIST data sets for the Chinese-English translation task. Experimental results show that our method can effectively adapt log-linear model parameters to different test data, and consistently achieves good translation performance compared with conventional methods that use a fixed model parameter setting across different data sets.

1 Introduction

In recent years, log-linear model (Och and Ney, 2002) has been a mainstream method to formulate statistical models for machine translation. Using this formulation, various kinds of relevant properties and data statistics used in the translation process, either on the monolingual-side or on the bilingual-side, are encoded and used as real-valued *feature functions*, thus it provides an effective mathematical framework to accommodate a large variety of SMT formalisms with different computational linguistic motivations.

*This work was done while the author was visiting Microsoft Research Asia.

Formally, in a log-linear SMT model, given a source sentence f, we are to find a translation e^* with largest posterior probability among all possible translations:

$$e^* = \operatorname*{argmax}_e \Pr(e|f)$$

and the posterior probability distribution $\Pr(e|f)$ is directly approximated by a log-linear formulation:

$$\Pr(e|f) = p_{\boldsymbol{\lambda}}(e|f) = \frac{\exp(\sum_{m=1}^{M} \lambda_m h_m(e,f))}{\sum_{e'} \exp(\sum_{m=1}^{M} \lambda_m h_m(e',f))} \quad (1)$$

in which h_m's are feature functions and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M)$ are model parameters (*feature weights*).

For a successful practical log-linear SMT model, it is usually a combined result of the several efforts:

- Construction of well-motivated SMT models
- Accurate estimation of feature functions
- Appropriate scaling of log-linear model features (feature weight tuning).

In this paper, we focus on the last mentioned issue – parameter tuning for log-linear model. In general, log-linear model parameters are optimized on a held-out development data set. Using this method, similarly to many machine learning tasks, the model parameters are solely tuned based on the development data, and the optimality of obtained model on unseen test data relies on the assumption that both development and test data observe identical probabilistic distribution,

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 662–670,
Beijing, August 2010

which often does not hold for real-world data. The goal of this paper is to investigate novel methods for test data dependent model parameter selection. We begin with discussing the principle of parameter learning for log-linear SMT models, and explain the rationale of task transformation from parameter selection to development data selection. We describe two algorithms for automatic development set construction, and evaluated our method on several NIST MT evaluation data sets. Experimental results show that our method can effectively adapt log-linear model parameters to different test data and achieves consistent good translation performance compared with conventional methods that use a group of fixed model parameters across different data sets.

2 Model Learning for SMT with Log-linear Models

Model learning refers to the task to estimate a group of suitable log-linear model parameters $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M)$ for use in Equation 1, which is often formulated as an optimization problem that finds the parameters maximizing certain *goodness* of the translations generated by the learnt model on a development corpus D. The goodness can be measured with either the translations' likelihood or specific machine translation evaluation metrics such as TER or BLEU.

More specifically, let e^* be the most probable translation of D with respect to model parameters $\boldsymbol{\lambda}$, and $E(e^*, \boldsymbol{\lambda}, D)$ be a score function indicating the goodness of translation e^*, then a parameter estimation algorithm will try to find the $\boldsymbol{\lambda}$ which satisfies:

$$\boldsymbol{\lambda}^* = \operatorname*{argmax}_{\boldsymbol{\lambda}} E(e^*, \boldsymbol{\lambda}, D) \qquad (2)$$

Note when the goodness scoring function $E(\cdot)$ is specified, the parameter learning criterion in Equation 2 indicates that the derivation of model parameters $\boldsymbol{\lambda}^*$ only depends on development data D, and does not require any knowledge of test data T. The underlying rationale for this rule is that if the test data T observes the same distribution as D, $\boldsymbol{\lambda}^*$ will be optimal for both of them.

On the other side, however, when there are mismatches between development and test data, the translation performance on test data will be sub-optimal, which is very common for real-world data. Due to the difference between data sets, generally there is no such $\boldsymbol{\lambda}^*$ that is optimal for multiple data sets at the same time. Table 1 shows some empirical evidences when two data sets are mutually used as development and test data. In this setting, we used a hierarchical phrase based decoder and 2 years' evaluation data of NIST Chinese-to-English machine translation task (for the year 2008 only the newswire subset was used because we want to limit both data sets within the same domain to show that data mismatch also exists even if there is no domain difference), and report results using BLEU scores. Model parameters were tuned using the MERT algorithm (Och, 2003) optimized for BLEU metric.

Dev data	MT05	MT08-nw
MT05	**0.402**	0.306
MT08-nw	0.372	**0.343**

Table 1: Translation performance of cross development/test on two NIST evaluation data sets.

In our work, we present a solution to this problem by using test data dependent model parameters for test data translation. As discussed above, since model parameters are solely determined by development data D, selection of log-linear model parameters is basically equivalent to selecting a set of development data D.

However, automatic development data selection in current SMT research remains a relatively open issue. Manual selection based on human experience and observation is still a common practice.

3 Adaptive Model Parameter Selection

An important heuristic behind manual development data selection is to use the dataset which is as similar to test set as possible in order to work around the data mismatch problem to maximal extent. There are also empirical evidences supporting this heuristics. For instance, it is generally perceived that data set MT03 is more similar to MT05, while MT06-nw is closer to MT08-nw. Table 2 shows experimental results using model parameters induced from MT03 and MT06-nw as development sets with the same settings as in Table 1. As expected, MT06-nw is far more suitable than MT03 as the development data for MT08-nw; yet for test set MT05, the situation is just the opposite.

Dev data	MT05	MT08-nw
MT03	**0.397**	0.306
MT06-nw	0.381	**0.337**

Table 2: Translation performance on different test sets of using different development sets.

In this work, this heuristic is further exploited for automatic development data selection when there is no prior knowledge of the test data available. In the following discussion, we assume the availability of a set of candidate source sentences together with translation references that are qualified for the log-linear model parameter learning task. Let D_F be the full candidate set, given a test set T, the task of selecting a set of development data which can optimize the translation quality on T can be transformed to searching for a suitable subset of D_F which is most similar to T:

$$D^* = \underset{D \subseteq D_F}{\operatorname{argmax}} \operatorname{Sim}(D, T)$$

To achieve this goal, we need to address the following key issues:

- How to define and compute $\operatorname{Sim}(D, T)$, the similarity between different data sets;
- How to extract development data sets from a full candidate set for unseen test data.

3.1 Dataset Similarity

Computing document similarity is a classical task in many research areas such as information retrieval and document classification. However, typical methods for computing document similarity may not be suitable for our purpose. The reasons are two-fold:

1. The sizes of both development and test data are small in usual circumstances, and using similarity measures such as cosine or dice coefficient based on term vectors will suffer from severe data sparseness problems. As a result, the obtained similarity measure will not be statistically reliable.
2. More importantly, what we care about here is not the surface string similarity. Instead, we need a method to measure how similar two data sets are from the view of a log-linear SMT model.

Next we start with discussing the similarity between sentences. Given a source sentence f, we denote its possible translation space with $\mathcal{H}(f)$. In a log-linear SMT model, every translation $e \in \mathcal{H}(f)$ is essentially a feature vector $\boldsymbol{h}(e) = (h_1, \ldots, h_M)$. Accordingly, the similarity between two sentences f_1 and f_2 should be defined on the feature space of the model in use. Let $\boldsymbol{V}(f) = \{\boldsymbol{h}(e) : e \in \mathcal{H}(f)\}$ be the set of feature vectors for all translations in $\mathcal{H}(f)$, we have

$$\operatorname{Sim}(f_1, f_2) = \operatorname{Sim}\Big(\boldsymbol{V}(f_1), \boldsymbol{V}(f_2)\Big) \quad (3)$$

Because it is not practical to compute Equation 3 directly by enumerating all translations in $\mathcal{H}(f_1)$ and $\mathcal{H}(f_2)$ due to the huge search space in SMT tasks, we need to resort to some approximations. A viable solution to this is that if we can use a single feature vector $\tilde{\boldsymbol{h}}(f)$ to represent $\boldsymbol{V}(f)$, then Equation 3 can be simply computed using existing vector similarity measures.

One reasonable method to derive $\tilde{\boldsymbol{h}}(f)$ is to use a feature vector based on the *average* principle – each dimension of the vector is set to the expectation of its corresponding feature value over all translations:

$$\tilde{\boldsymbol{h}}(f) = \sum_{e \in \mathcal{H}(f)} P(e|f)\boldsymbol{h}(e) \quad (4)$$

An alternative and much simpler way to compute $\tilde{\boldsymbol{h}}(f)$ is to employ the *max* principle in which we just use the feature vector of the best translation in $\mathcal{H}(f)$:

$$\tilde{\boldsymbol{h}}(f) = \boldsymbol{h}(e^*) \quad (5)$$

where $e^* = \operatorname{argmax}_e P(e|f)$.

Note that in both Equation 4 and Equation 5 we make use of e's posterior probability $P(e|f)$.

Since the true distribution is unknown, a pre-learnt model $\mathcal{M}$ has to be used to assign approximate probabilities to translations, which indicates that the obtained similarity depends on a specific model. As a convention, we use $\mathrm{Sim}_{\mathcal{M}}(f_1, f_2)$ to denote the similarity between f_1 and f_2 based on $\mathcal{M}$, and call $\mathcal{M}$ the reference model of the computed similarity. To avoid unexpected bias caused by a single reference model, multiple reference models can be simultaneously used, and the similarity is defined to be the maximum of all model-dependent similarity values:

$$\mathrm{Sim}(f_1, f_2) = \max_{\mathcal{M}} \mathrm{Sim}_{\mathcal{M}}(f_1, f_2) \tag{6}$$

where $\mathcal{M}$ belongs to $\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$, which is the set of reference models under consideration.

To generalize this method to data set level, we compute the vector $\tilde{\boldsymbol{h}}(S)$ for a data set $S = (f_1, \ldots, f_{|S|})$ as follows:

$$\tilde{\boldsymbol{h}}(S) = \sum_{i=1}^{|S|} \tilde{\boldsymbol{h}}(f_i) \tag{7}$$

3.2 Development Sets Pre-construction

In the following, we sketch a method for automatically building a set of development data based on the full candidate set D_F before seeing any test data.

Theoretically, a subset of D_F containing randomly sampled sentences from D_F will not meet our requirement well because it is very probable that it will observe a distribution similar to D_F. What we expect is that the pre-built development sets can approximate as many as possible typical data distributions that can be estimated from subsets of D_F. Our solution is based on the assumption that D_F can be depicted by some mixture models, hence we can use classical clustering methods such as k-means to partition D_F into subsets with different distributions.

Let S_F be the set of extracted development data from D_F. The construction of S_{D_F} proceeds as following:

1. Train a log-linear model $\mathcal{M}_F$ using D_F as development data;
2. Compute a feature vector $\tilde{\boldsymbol{h}}(d)$[1] for each sentence $d \in D_F$ using $\mathcal{M}_F$ as reference model;
3. Cluster sentences in D_F using $\tilde{\boldsymbol{h}}(d)/|d|$ as feature vectors;
4. Add obtained sentence clusters to S_{D_F} as candidate development sets.

In the third step, since the feature vector $\tilde{\boldsymbol{h}}(d)$ is defined at sentence level, it is averaged by the number of words in d so that it is irrelevant to the length of a sentence. Considering the outputs of unsupervised data clustering methods are usually sensitive to initial conditions, we include in S_{D_F} sentence clusters based on different initialization configurations to remove related random effects. An initialization configuration for sentence clustering in our work includes starting point for each cluster and total number of clusters. In fact, the inclusion of more sentence clusters increases the diversity of the resulted S_{D_F} as well.

At decoding time, when a test set T is presented, we compute the similarity between T and each development set $D \in S_{D_F}$, and choose the one with largest similarity score as the development set for T:

$$D^* = \operatorname*{argmax}_{D \in S_{D_F}} \mathrm{Sim}(T, D) \tag{8}$$

When a single reference model is used to compute $\mathrm{Sim}(T, D)$, $\mathcal{M}_F$ is a natural choice. In the multi-model setting as shown in Equation 6, models learnt from the development sets in S_{D_F} can serve this purpose.

Note in this method model learning is not required for every new test set because the model parameters for each development set in S_{D_F} can also be pre-learnt and ready to be used for decoding.

3.3 Dynamic Development Set Construction

In the previous method, test data T is only involved in the process of choosing a development set from a list of candidates but not in process of development set construction. Next we present a

[1]Throughout this paper, a development sentence d generally refers to the source part of it if there is no extra explanation.

method for building a development set on demand based on test data T.

Let $D_F = (d_1, \ldots, d_n)$ be the data set containing all candidate sentences for development data selection. The method is iterative process in which development data and learnt model are alternatively updated. Detailed steps are illustrated as follows:

1. Let $i = 0$, $D_0 = D_F$;
2. Train a model $\mathcal{M}_i$ based on D_i;
3. For each $d_k \in D_F$, compute the similarity score $\mathrm{Sim}_{\mathcal{M}_i}(T, d_k)$ between T and d_k based on model $\mathcal{M}_i$;
4. Select top n candidate sentences with highest similarity scores from D_F to form D_{i+1};
5. Repeat step 2 to step 4 until the similarity between T and latest selected development data converges (the increase in similarity measure is less than a specified threshold compared to last round) or the specified iteration limit is reached.

In step 4, D_{i+1} is greedily extracted from D_F, and there is no guarantee that $\mathrm{Sim}_{\mathcal{M}_i}(T, D_{i+1})$ will increase or decrease after a new sentence is added to D_{i+1}. Thereby the number of selected sentences n needs to be empirically determined. If n is too small, neither the selected data nor the learnt model parameters will be statistically reliable; while if n is too large, we may have to include some sentences that are not suitable for test data in the development data, and miss the opportunity to extract the most desirable development set.

One drawback of this method is the relatively high computational cost because it requires multiple parameter training passes when any test set is presented to the system for translation.

4 Experiments

4.1 Data

Experiments were conducted on the data sets used for NIST Chinese-English machine translation evaluation tasks. MT03 and MT06 data sets, which contain 919 and 1,664 sentences respectively, were used for development data in various settings. MT04, MT05 and MT08 data sets were used for test purpose. In some settings, we also used a test set MT0x, which containing 1,000 sentences randomly sampled from the above 3 data sets. All the translation performance results were measured in terms of case-insensitive BLEU scores.

For all experiments, all parallel corpora available to the constrained track of NIST 2008 Chinese-English MT evaluation task were used for translation model training, which consist of around 5.1M bilingual sentence pairs. GIZA++ was used for word alignment in both directions, which was further refined with the intersec-diag-grow heuristics.

We used a 5-gram language model which was trained from the Xinhua portion of English Gigaword corpus version 3.0 from LDC and the English part of parallel corpora.

4.2 Machine Translation System

We used an in-house implementation of the hierarchical phrase-based decoder as described in Chiang (2005). In addtion to the standard features used in Chiang (2005), we also used a lexicon feature indicating how many word paris in the translation found in a conventional Chinese-English lexicon. Phrasal rules were extracted from all the parallel data, but hierarchical rules were only extracted from the FBIS part of the parallel data which contains around 128,000 sentence pairs. For all the development data, feature weights of the decoder were tuned using the MERT algorithm (Och, 2003).

4.3 Results of Development Data Pre-construction

In the following we first present some overall results using the method of development data pre-construction, then dive into more detailed settings of the experiments.

Table 3 shows the results using 3 different data sets for log-linear model parameter tuning. Elements in the first column indicate the data sets used for parameter tuning, and other columns contain evaluation results on different test sets. In the

Tuning set	MT04	MT05	MT08	MT0x
MT03	0.399 / 0.392	0.395 / 0.390	0.241 / 0.258	0.319 / 0.322
MT06	0.381 / 0.388	0.382 / 0.391	0.275 / 0.283	0.343 / 0.342
MT03+MT06	0.391 / 0.401	0.392 / 0.397	0.265 / 0.281	0.336 / 0.345
Oracle cluster	0.401	0.398	0.293	0.345
Self-training	0.406	0.402	0.298	0.351

Table 3: Translation performance using different methods and data sets for parameter tuning.

third row of the table, MT03+MT06 means combining the data sets of MT03 and MT06 together to form a larger tuning set. The first number in each cell denotes the BLEU score using the tuning set as standard development set D, and the second for using the tuning set as a candidate set D_F.

For all experiment settings in the table, we used cosine value between feature vectors to measure similarity between data sets, and feature vectors were computed according to Equation 5 and Equation 7 using a reference model which is trained on the corresponding candidate set D_F as development set.[2] We adopted the k-means algorithm for data clustering with the number of clusters iterating from 2 to 5. In each iteration, we ran 4 passes of clustering using different initial values. Therefore, in total there are 56 sentence clusters generated in each S_{D_F}.[3]

From the table it can be seen that given the same set of sentences (MT03, MT06 and MT03+MT06), when they are used as the candidate set D_F for the development set preconstruction method, the translation performance is generally better than when they are just used as development sets as a whole. Using MT03 data set as D_F is an exception: there is slight performance drop on test sets MT04 and MT05, but it also helps reduce the performance *see-saw* problem on different test sets as shown in Table 1. Meanwhile, in the other two settings of D_F, we observed significant BLEU score increase on all test sets but MT0x (on which the performance almost kept unchanged). In addition, the fact that using MT03+MT06 as D_F achieves best (or almost best) performance on all test sets implies that it should be a better choice to include as diverse data as possible in D_F.

[2]For example, in all the experiments in the row of MT03 as D_F, we use the same reference model trained with MT03 as development set.

[3]Sometimes some clusters are empty or contain too few sentences, so the actual number may be smaller.

We also appended two oracle BLEU numbers for each test set in Table 3 for reference. One is denoted with *oracle cluster*, which is the highest possible BLEU that can be achieved on the test set when the development set must be chosen from the sentence clusters in $S_{MT03+MT06}$. The other is labeled as *self-training*, which is the BLEU score that can be obtained when the test data itself is used as development data. This number can serve as actual performance upper bound on the test set.

Next we investigated the impact of using different ways to compute feature vectors presented in Section 3.1. We re-ran some previous experiments on test sets MT04, MT05 and MT08 using MT03+MT06 as D_F. Most settings were kept unchanged except that the feature vector of each sentence was computed according to Equation 4. A 20-best translation list was used to approximate $\mathcal{H}(f)$. The results are shown in Table 4.

Test set	average	max
MT04	0.397	0.401
MT05	0.393	0.397
MT08	0.286	0.281

Table 4: Translation performance when using averaged feature values for similarity computation.

The numbers in the second column are based on Equation 4. Numbers based on Equation 5 are also listed in the third column for comparison. In all the experiment settings we did not observe consistent or significant advantage when using Equation 4 over using Equation 5. Since Equation 5 is much simpler, it is a good decision to use it in practice. So did we conduct all following experiments based on Equation 5.

We are also interested in the correlation between two measures: the similarity between development and test data and the actual translation performance on test data.

First we would like to echo the motivating experiment presented in Section 3. Table 5 shows the similarity between the data sets used in the experiment with $\mathcal{M}_{\mathrm{MT03+MT06}}$ as reference model. Obviously the results in Table 2 and Table 5 fit each other very well.

Dev data	MT05	MT08-nw
MT03	**0.99988**	0.99012
MT06-nw	0.99004	**0.99728**

Table 5: Similarity between NIST data sets.

Figure 1 shows the results of a set of more comprehensive experiments on MT05 data set concerning the similarity between development and test sets.

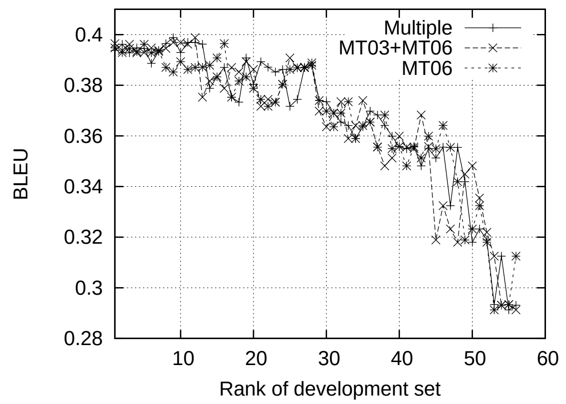


Figure 1: Correlation between similarity and BLEU on MT05 data set

In the figure, every data line shows how BLEU score changes when different pre-built development set in $S_{MT03+MT06}$ is used for model learning. The data points in each line are sorted by the rank of similarity between the development set in use and the MT05 data set. We also compared results based on 3 reference model settings. In the first one (multiple), the similarity was computed using Equation 6, and the reference model set contains all models learnt from the development sets in $S_{MT03+MT06}$. The other two settings use reference models learnt from MT06 and MT03+MT06 data sets respectively.

We can observe from the figure that the correlation between BLEU scores and data set similarity can only be identified on macro scales for all the three similarity settings. Although using data similarity may not be able to select the perfect development data set from S_{D_F}, by picking a development set with highest similarity score, we can usually (almost always) get good enough BLEU scores in our experiments.

4.4 Results of Development Data Dynamic Generation

We ran two sets of experiments for the method of development data dynamic construction.

The first one was designed to investigate how the size of extracted development data affects the translation performance. Using MT05 and MT08 as test sets and MT03+MT06 as D_F, we ran experiments for the algorithm presented in Section 3.3 with $n = 200$ to $n = 1,000$. In this experiment we did not observe significant enough changes in BLEU scores – the difference between the highest and lowest numbers is generally less than 0.005.

The second one aimed at examining how BLEU numbers changes when the extracted development data were iteratively updated. Figure 2 shows one set of results on test sets MT05 and MT08 using MT03+MT06 data set as D_F and n set to 400.

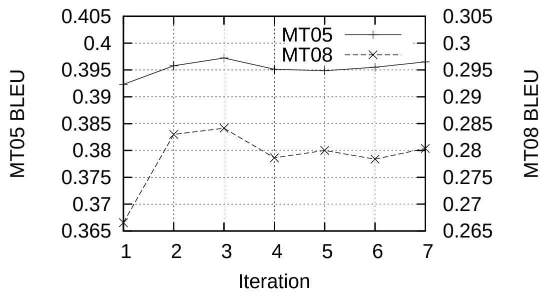


Figure 2: BLEU score as function of iteration in dynamic development data extraction.

The similarity usually converged after 2 to 3 iterations, which is consistent with trend of BLEU scores on test sets. However, in all our experimental settings, we did not observe any results significantly better than using the development set pre-construction method.

5 Discussions

Some of the previous work related to building adaptive SMT systems were discussed in the domain adaptation context, in which one fundamental idea is to estimate a more suitable domain-specific translation model or language model. When the target domain is already known, adding a small amount of domain data (both monolingual and bilingual) to the existing training corpora has been shown to be very effective in practice. But model adaptation is required in more scenarios other than explicitly defined domains. As shown by the results in Table 2, even for the data from the same domain, distribution mismatch can also be a problem.

There are also considerable efforts made to deal with the unknown distribution of text to be translated, and the research topics were still focused on translation and language model adaptation. Typical methods used in this direction include dynamic data selection (Lü et al., 2007; Zhao et al., 2004; Hildebrand et al., 1995) and data weighting (Foster and Kuhn, 2007; Matsoukas et al., 2009). All the mentioned methods use information retrieval techniques to identify relevant training data from the entire training corpora.

Our work presented here also makes no assumption about the distribution of test data, but it differs from the previous methods significantly from a log-linear model's perspective. Adjusting translation and language models based on test data can be viewed as *adaptation of feature values*, while our method is essentially *adaptation of feature weights*. This difference makes these two kinds of methods complementary to each other — it is possible to make further improvement by using both of them in one task.

To our knowledge, there is no dedicated discussion on principled methods to perform development data selection in previous research. In Lü et al. (2007), log-linear model parameters can also be adjusted at decoding time. But in their approach, the adjustment was based on heuristic rules and re-weighted training data distribution. In addition, compared with training data selection, the computational cost of development data selection is much smaller.

From machine learning perspective, both proposed methods can be viewed as certain form of transductive learning applied to the SMT task (Ueffing et al., 2007). But our methods do not rely on surface similarities between training and training/development sentences, and development/test sentences are not used to re-train SMT sub-models.

6 Conclusions and Future Work

In this paper, we addressed the data mismatch issue between training and decoding time of log-linear SMT models, and presented principled methods for dynamically inferring test data dependent model parameters with development set selection. We describe two algorithms for this task, development set pre-construction and dynamic construction, and evaluated our method on the NIST data sets for the Chinese-English translation task. Experimental results show that our methods are capable of consistently achieving good translation performance on multiple test sets with different data distributions without manual tweaking of log-linear model parameters. Though theoretically using the dynamic construction method could bring better results, the pre-construction method performs comparably well in our experimental settings. Considering the fact that the pre-consruction method is computationally cheaper, it should be a better choice in practice.

In the future, we are interested in two directions. One is to explore the possibility to perform data clustering on test set as well and choosing suitable model parameters for each cluster separately. The other involves dynamic SMT model selection – for example, some parts of the test data fit the phrase-based model better while other parts can be better translated using a syntax-based model.

References

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. of the 43th Annual Meeting of the Association for Computational Linguistic (ACL)*. Ann Arbor, Michigan.

George Foster and Roland Kuhn. 2007. Mixture-Model Adaptation for SMT. In *Proc. of the Second ACL Workshop on Statistical Machine Translation.*. Prague, Czech Republic.

Michel Galley, Mark Hopkins, Kevin Knight and Daniel Marcu. 2004. What's in a translation rule? In *Proc. of the Human Language Technology Conf. (HLT-NAACL)*. Boston, Massachusetts.

Almut Hildebrand, Matthias Eck, Stephan Vogel, and Alex Waibel. 1995. Adaptation of the Translation Model for Statistical Machine translation Based on Information Retrieval. In *Proc. of EAMT*. Budapest, Hungary.

Philipp Koehn, Franz Och and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proc. of the Human Language Technology Conf. (HLT-NAACL)*. Edmonton, Canada.

Yang Liu, Qun Liu, and Shouxun Lin. 2007. Tree-to-string alignment template for statistical machine translation. In *Proc. of the 45th Annual Meeting of the Association for Computational Linguistic (ACL)*. Prague, Czech Republic.

Yajuan Lü, Jin Huang and Qun Liu. 2007. Improving Statistical Machine Translation Performance by Training Data Selection and Optimization. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*. Prague, Czech Republic.

Spyros Matsoukas, Antti-Veikko I. Rosti and Bing Zhang. 2009. Discriminative Corpus Weight Estimation for Machine Translation. In *Proc. of the Conference on Empirical Methods in Natural Language Processing*. Singapore.

Franz Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translation. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistic (ACL)*. Philadelphia, PA.

Franz Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proc. of the 41th Annual Meeting of the Association for Computational Linguistic (ACL)*. Sapporo, Japan.

Nicola Ueffing, Gholamreza Haffari and Anoop Sarkar. 2007. Transductive learning for statistical machine translation. In *Proc. of the Annual Meeting of the Association for Computational Linguistics*. Prague, Czech Republic.

Bing Zhao, Matthias Eck, and Stephan Vogel. 2004. Language Model Adaptation for Statistical Machine Translation with Structured Query Models. In *Proc. of COLING*. Geneva, Switzerland.

Learning the Scope of Negation via Shallow Semantic Parsing

Junhui Li   Guodong Zhou[*]   Hongling Wang   Qiaoming Zhu
School of Computer Science and Technology
Soochow University at Suzhou
{lijunhui, gdzhou, redleaf, qmzhu}@suda.edu.cn

Abstract

In this paper we present a simplified shallow semantic parsing approach to learning the scope of negation (SoN). This is done by formulating it as a shallow semantic parsing problem with the negation signal as the predicate and the negation scope as its arguments. Our parsing approach to SoN learning differs from the state-of-the-art chunking ones in two aspects. First, we extend SoN learning from the chunking level to the parse tree level, where structured syntactic information is available. Second, we focus on determining whether a constituent, rather than a word, is negated or not, via a simplified shallow semantic parsing framework. Evaluation on the BioScope corpus shows that structured syntactic information is effective in capturing the domination relationship between a negation signal and its dominated arguments. It also shows that our parsing approach much outperforms the state-of-the-art chunking ones.

1 Introduction

Whereas negation in predicate logic is well-defined and syntactically simple, negation in natural language is much complex. Generally, learning the scope of negation involves two subtasks: negation signal finding and negation scope finding. The former decides whether the words in a sentence are negation signals (i.e., words indicating negation, e.g., *no*, *not*, *fail*, *rather than*), where the semantic information of the words, rather than the syntactic information, plays a critical role. The latter determines the sequences of words in the sentence which are negated by the given negation signal. Compared with negation scope finding, negation signal finding is much simpler and has been well resolved in the literature, e.g. with the accuracy of 95.8%-98.7% on the three subcorpora of the Bioscope corpus (Morante and Daelemans, 2009). In this paper, we focus on negation scope finding instead. That is, we assume golden negation signal finding.

Finding negative assertions is essential in information extraction (IE), where in general, the aim is to derive factual knowledge from free text. For example, Vincze et al. (2008) pointed out that the extracted information within the scopes of negation signals should either be discarded or presented separately from factual information. This is especially important in the biomedical domain, where various linguistic forms are used extensively to express impressions, hypothesized explanations of experimental results or negative findings. Szarvas et al. (2008) reported that 13.45% of the sentences in the abstracts subcorpus of the BioScope corpus and 12.70% of the sentences in the full papers subcorpus of the Bioscope corpus contain negative assertions. In addition to the IE tasks in the biomedical domain, SoN learning has attracted more and more attention in some natural language processing (NLP) tasks, such as sentiment classification (Turney, 2002). For example, in the sentence "*The chair is not comfortable but cheap*", although both the polarities of the words "*comfortable*" and "*cheap*" are positive, the polarity of "the chair" regarding the attribute "*cheap*" keeps positive while the polarity of "the chair" regarding the attribute "*comfortable*" is reversed due to the negation signal "*not*".

Most of the initial research on SoN learning focused on negated terms finding, using either some heuristic rules (e.g., regular expression), or machine learning methods (Chapman et al., 2001; Huang and Lowe, 2007; Goldin and Chapman, 2003). Negation scope finding has been largely ignored until the recent release of

[*] Corresponding author

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 671–679,
Beijing, August 2010

the BioScope corpus (Szarvas et al., 2008; Vincze et al., 2008). Morante et al. (2008) and Morante and Daelemans (2009) pioneered the research on negation scope finding by formulating it as a chunking problem, which classifies the words of a sentence as being inside or outside the scope of a negation signal. However, this chunking approach suffers from low performance, in particular on long sentences, due to ignoring structured syntactic information. For example, given golden negation signals on the Bioscope corpus, Morante and Daelemans (2009) only got the performance of 50.26% in PCS (percentage of correct scope) measure on the full papers subcorpus (22.8 words per sentence on average), compared to 87.27% in PCS measure on the clinical reports subcorpus (6.6 words per sentence on average).

This paper explores negation scope finding from a parse tree perspective and formulates it as a shallow semantic parsing problem, which has been extensively studied in the past few years (Carreras and Màrquez, 2005). In particular, the negation signal is recast as the predicate and the negation scope is recast as its arguments. The motivation behind is that structured syntactic information plays a critical role in negation scope finding and should be paid much more attention, as indicated by previous studies in shallow semantic parsing (Gildea and Palmer, 2002; Punyakanok et al., 2005). Our parsing approach to negation scope finding differs from the state-of-the-art chunking ones in two aspects. First, we extend negation scope finding from the chunking level into the parse tree level, where structured syntactic information is available. Second, we focus on determining whether a constituent, rather than a word, is negated or not. Evaluation on the BioScope corpus shows that our parsing approach much outperforms the state-of-the-art chunking ones.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the Bioscope corpus on which our approach is evaluated. Section 4 describes our parsing approach by formulating negation scope finding as a simplified shallow semantic parsing problem. Section 5 presents the experimental results. Finally, Section 6 concludes the work.

2 Related Work

While there is a certain amount of literature within the NLP community on negated terms finding (Chapman et al., 2001; Huang and Lowe, 2007; Goldin and Chapman, 2003), there are only a few studies on negation scope finding (Morante et al., 2008; Morante and Daelemans, 2009).

Negated terms finding

Rule-based methods dominated the initial research on negated terms finding. As a representative, Chapman et al. (2001) developed a simple regular expression-based algorithm to detect negation signals and identify medical terms which fall within the negation scope. They found that their simple regular expression-based algorithm can effectively identify a large portion of the pertinent negative statements from discharge summaries on determining whether a finding or disease is absent. Besides, Huang and Lowe (2007) first proposed some heuristic rules from a parse tree perspective to identify negation signals, taking advantage of syntactic parsing, and then located negated terms in the parse tree using a corresponding negation grammar.

As an alternative to the rule-based methods, various machine learning methods have been proposed for finding negated terms. As a representative, Goldin and Chapman (2003) adopted both Naïve Bayes and decision trees to distinguish whether an observation is negated by the negation signal "*not*" in hospital reports.

Negation scope finding

Morante et al. (2008) pioneered the research on negation scope finding, largely due to the availability of a large-scale annotated corpus, the Bioscope corpus. They approached the negation scope finding task as a chunking problem which predicts whether a word in the sentence is inside or outside of the negation scope, with proper post-processing to ensure consecutiveness of the negation scope. Morante and Daelemans (2009) further improved the performance by combing several classifiers.

Similar to SoN learning, there are some efforts in the NLP community on learning the scope of speculation. As a representative, Özgür and Radev (2009) divided speculation learning into two subtasks: speculation signal finding and speculation scope finding. In particular, they formulated speculation signal finding as a classification problem while employing some heuristic rules from the parse tree perspective on speculation scope finding.

3 Negation in the BioScope Corpus

This paper employs the BioScope corpus (Szarvas et al., 2008; Vincze et al., 2008)[1], a freely downloadable negation resource from the biomedical domain, as the benchmark corpus. In this corpus, every sentence is annotated with negation signals and speculation signals (if it has), as well as their linguistic scopes. Figure 1 shows a self-explainable example. In this paper, we only consider negation signals, rather than speculation ones. Our statistics shows that 96.57%, 3.23% and 0.20% of negation signals are represented by one word, two words and three or more words, respectively. Additional, adverbs (e.g., *not*, *never*) and determiners (e.g., *no*, *neither*) occupy 45.66% and 30.99% of negation signals, respectively.

<sentence id="S26.8">These findings <xcope id="X26.8.2"><cue type="**speculation**" ref="X26.8.2">indicate that</cue> <xcope id="X26.8.1">corticosteroid resistance in bronchial asthma <cue type="**negation**" ref="X26.8.1">can not</cue> be explained by abnormalities in corticosteroid receptor characteristics</xcope></xcope>.</sentence>

Figure 1: An annotated sentence in the BioScope corpus.

The Bioscope corpus consists of three subcorpora: the full papers and the abstracts from the GENIA corpus (Collier et al., 1999), and clinical (radiology) reports. Among them, the full papers subcorpus and the abstracts subcorpus come from the same genre, and thus share some common characteristics in statistics, such as the number of words in the negation scope to the right (or left) of the negation signal and the average scope length. In comparison, the clinical reports subcorpus consists of clinical radiology reports with short sentences. For detailed statistics about the three subcorpora, please see Morante and Daelemans (2009).

For preprocessing, all the sentences in the Bioscope corpus are tokenized and then parsed using the Berkeley parser[2] (Petrov and Klein, 2007) trained on the GENIA TreeBank (GTB) 1.0 (Tateisi et al., 2005)[3], which is a bracketed corpus in (almost) PTB style. 10-fold cross-validation on GTB1.0 shows that the parser achieves the performance of 86.57 in F1-measure. It is worth noting that the GTB1.0 corpus includes all the sentences in the abstracts subcorpus of the Bioscope corpus.

4 Negation Scope Finding via Shallow Semantic Parsing

In this section, we first formulate the negation scope finding task as a shallow semantic parsing problem. Then, we deal with it using a simplified shallow semantic parsing framework.

4.1 Formulating Negation Scope Finding as a Shallow Semantic Parsing Problem

Given a parse tree and a predicate in it, shallow semantic parsing recognizes and maps all the constituents in the sentence into their corresponding semantic arguments (roles) of the predicate. As far as negation scope finding considered, the negation signal can be regarded as the predicate[4], while the scope of the negation signal can be mapped into several constituents which are negated and thus can be regarded as the arguments of the negation signal. In particular, given a negation signal and its negation scope which covers $word_m$, …, $word_n$, we adopt the following two heuristic rules to map the negation scope of the negation signal into several constituents which can be deemed as its arguments in the given parse tree.

1) The negation signal itself and all of its ancestral constituents are non-arguments.
2) If constituent X is an argument of the given negation signal, then X should be the highest constituent dominated by the scope of $word_m$, …, $word_n$. That is to say, X's parent constituent must cross-bracket or include the scope of $word_m$, …, $word_n$.

[1] http://www.inf.u-szeged.hu/rgai/bioscope

[2] http://code.google.com/p/berkeleyparser/

[3] http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA

[4] If a negation signal consists of multiply words (e.g., rather than), the last word (e.g., than) is chosen to represent the negation signal.

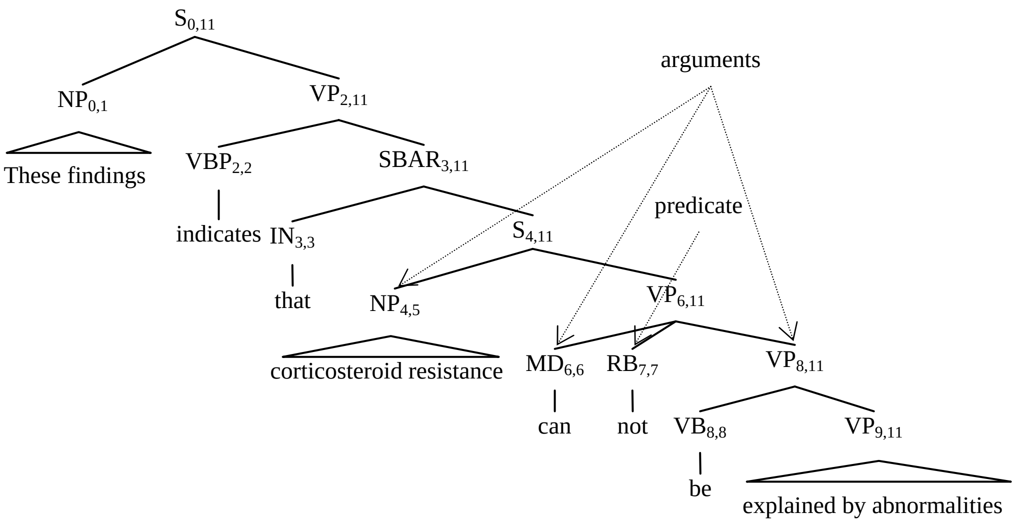


Figure 2: An illustration of a negation signal and its arguments in a parse tree.

The first rule ensures that no argument covers the negation signal while the second rule ensures no overlap between any two arguments. For example, in the sentence "*These findings indicate that corticosteroid resistance can not be explained by abnormalities*", the negation signal "*can not*" has the negation scope "*corticosteroid resistance can not be explained by abnormalities*". As shown in Figure 2, the node "$RB_{7,7}$" (i.e., *not*) represents the negation signal "*can not*" while its arguments include three constituents {$NP_{4,5}$, $MD_{6,6}$, and $VP_{8,11}$}. It is worth noting that according to the above rules, negation scope finding via shallow semantic parsing, i.e. determining the arguments of a given negation signal, is robust to some variations in parse trees. This is also empirically justified by our later experiments. For example, if the $VP_{6,11}$ in Figure 2 is incorrectly expanded by the rule $VP_{6,11} \rightarrow MD_{6,6}+RB_{7,7}+VB_{8,8}+VP_{9,11}$, the negation scope of the negation signal "*can not*" can still be correctly detected as long as {$NP_{4,5}$, $MD_{6,6}$, $VB_{8,8}$, and $VP_{9,11}$} are predicted as the arguments of the negation signal "*can not*".

Compared with common shallow semantic parsing which needs to assign an argument with a semantic label, negation scope finding does not involve semantic label classification and thus could be divided into three consequent phases: argument pruning, argument identification and post-processing.

4.2 Argument Pruning

Similar to the predicate-argument structures in common shallow semantic parsing, the negation signal-scope structures in negation scope finding can be also classified into several certain types and argument pruning can be done by employing several heuristic rules to filter out constituents, which are most likely non-arguments of a negation signal. Similar to the heuristic algorithm as proposed in Xue and Palmer (2004) for argument pruning in common shallow semantic parsing, the argument pruning algorithm adopted here starts from designating the negation signal as the current node and collects its siblings. It then iteratively moves one level up to the parent of the current node and collects its siblings. The algorithm ends when it reaches the root of the parse tree. To sum up, except the negation signal and its ancestral constituents, any constituent in the parse tree whose parent covers the given negation signal will be collected as argument candidates. Taking the negation signal node "$RB_{7,7}$" in Figure 2 as an example, constituents {$MD_{6,6}$, $VP_{8,11}$, $NP_{4,5}$, $IN_{3,3}$, $VBP_{2,2}$, and $NP_{0,1}$} are collected as its argument candidates consequently.

4.3 Argument Identification

Here, a binary classifier is applied to determine the argument candidates as either valid arguments or non-arguments. Similar to argument identification in common shallow semantic parsing, the structured syntactic information plays a critical role in negation scope finding.

Basic Features

Table 1 lists the basic features for argument identification. These features are also widely used in common shallow semantic parsing for both verbal and nominal predicates (Xue, 2008; Li et al., 2009).

Feature	Remarks
b1	Negation: the stem of the negation signal, e.g., not, rather_than. (*can_not*)
b2	Phrase Type: the syntactic category of the argument candidate. (*NP*)
b3	Path: the syntactic path from the argument candidate to the negation signal. (*NP<S>VP>RB*)
b4	Position: the positional relationship of the argument candidate with the negation signal. “left” or “right”. (*left*)

Table 1: Basic features and their instantiations for argument identification in negation scope finding, with $NP_{4,5}$ as the focus constituent (i.e., the argument candidate) and “*can not*” as the given negation signal, regarding Figure 2.

Additional Features

To capture more useful information in the negation signal-scope structures, we also explore various kinds of additional features. Table 2 shows the features in better capturing the details regarding the argument candidate and the negation signal. In particular, we categorize the additional features into three groups according to their relationship with the argument candidate (AC, in short) and the given negation signal (NS, in short).

Some features proposed above may not be effective in argument identification. Therefore, we adopt the greedy feature selection algorithm as described in Jiang and Ng (2006) to pick up positive features incrementally according to their contributions on the development data. The algorithm repeatedly selects one feature each time which contributes most, and stops when adding any of the remaining features fails to improve the performance. As far as the negation scope finding task concerned, the whole feature selection process could be done by first running the selection algorithm with the basic features (b1-b4) and then incrementally picking up effective features from (ac1-ac6, AC1-AC2, ns1-ns4, NS1-NS2, nsac1-nsac2, and NSAC1 -NSAC7).

Feature	Remarks
argument candidate (AC) related	
ac1	the headword (ac1H) and its POS (ac1P). (*resistance*, *NN*)
ac2	the left word (ac2W) and its POS (ac2P). (*that*, *IN*)
ac3	the right word (ac3W) and its POS (ac3P). (*can*, *MD*)
ac4	the phrase type of its left sibling (ac4L) and its right sibling (ac4R). (*NULL*, *VP*)
ac5	the phrase type of its parent node. (*S*)
ac6	the subcategory. (*S:NP+VP*)
combined features (AC1-AC2)	
b2&fc1H, b2&fc1P	
negation signal (NS) related	
ns1	its POS. (*RB*)
ns2	its left word (ns2L) and right word (ns2R). (*can*, *be*)
ns3	the subcategory. (*VP:MD+RB+VP*)
ns4	the phrase type of its parent node. (*VP*)
combined features (NS1-NS2)	
b1&ns2L, b1&ns2R	
NS-AC-related	
nsac1	the compressed path of b3: compressing sequences of identical labels into one. (*NP<S>VP>RB*)
nsac2	whether AC and NS are adjacent in position. “yes” or “no”. (*no*)
combined features (NSAC1-NSAC7)	
b1&b2, b1&b3, b1&nsac1, b3&NS1, b3&NS2, b4&NS1, b4&NS2	

Table 2: Additional features and their instantiations for argument identification in negation scope finding, with $NP_{4,5}$ as the focus constituent (i.e., the argument candidate) and “*can not*” as the given negation signal, regarding Figure 2.

4.4 Post-Processing

Although a negation signal in the BioScope corpus always has only one continuous block as its negation scope (including the negation signal itself), the negation scope finder may result in discontinuous negation scope due to independent prediction in the argument identification phase. Given the golden negation signals, we observed that 6.2% of the negation scopes predicted by our negation scope finder are discontinuous.

Figure 3 demonstrates the projection of all the argument candidates into the word level. According to our argument pruning algorithm in Section 4.2, except the words presented by the negation signal, the projection covers the whole sentence and each constituent (LAC_i or RAC_j in Figure 3) receives a probability distribution of being an argument of the given negation signal in the argument identification phase.

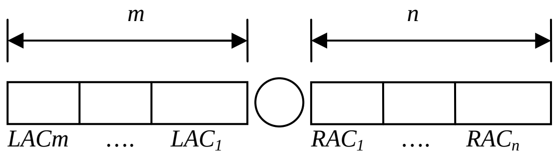


Figure 3: Projecting the left and the right argument candidates into the word level.

Since a negation signal is deemed inside of its negation scope in the BioScope corpus, our post-processing algorithm first includes the negation signal in its scope and then starts to identify the left and the right scope boundaries, respectively.

As shown in Figure 3, the left boundary has $m+1$ possibilities, namely the negation signal itself, the leftmost word of constituent LAC_i ($1<=i<=m$). Supposing LAC_i receives probability of P_i being an argument, we use the following formula to determine LAC_{k*} whose leftmost word represents the boundary of the left scope. If $k*=0$, then the negation signal itself represents its left boundary.

$$k^* = \arg\max_k \prod_{i=1}^{k} P_i * \prod_{i=k+1}^{m} (1-P_i)$$

Similarly, the right boundary of the given negation signal can be decided.

5 Experimentation

We have evaluated our shallow semantic parsing approach to negation scope finding on the BioScope corpus.

5.1 Experimental Settings

Following the experimental setting in Morante and Daelemans (2009), the abstracts subcorpus is randomly divided into 10 folds so as to perform 10-fold cross validation, while the performance on both the papers and clinical reports subcorpora is evaluated using the system trained on the whole abstracts subcorpus. In addition, SVMLight[5] is selected as our classifier. In particular, we adopt the linear kernel and the training parameter C is fine-tuned to 0.2.

The evaluation is made using the accuracy. We report the accuracy using three measures: *PCLB* and *PCRB*, which indicate the percentages of correct left boundary and right boundary respectively, *PCS*, which indicates the percentage of correct scope as a whole.

5.2 Experimental Results on Golden Parse Trees

In order to select beneficial features from the additional features proposed in Section 4.3, we randomly split the abstracts subcorpus into training and development datasets with proportion of 4:1. After performing the greedy feature selection algorithm on the development data, features {NSAC5, ns2R, NS1, ac1P, ns3, NSAC7, ac4R} are selected consecutively for argument identification. Table 3 presents the effect of selected features in an incremental way on the development data. It shows that the additional features significantly improve the performance by 11.66% in PCS measure from 74.93% to 86.59% ($\chi^2; p < 0.01$).

Feature	PCLB	PCRB	PCS
Baseline	84.26	88.92	74.93
+NSAC5	90.96	88.92	81.34
+ns2R	91.55	88.92	81.92
+NS1	92.42	89.50	83.09
+ac1P	93.59	89.50	84.26
+ns3	93.88	90.09	84.84
+NSAC7	94.75	89.80	85.42
+ac4R	95.04	90.67	86.59

Table 3: Performance improvement (%) of including the additional features in an incremental way on the development data (of the abstracts subcorpus).

However, Table 3 shows that the additional features behave quite differently in terms of PCLB and PCRB measures. For example, PCLB measure benefits more from features NSAC5, ns2R, NS1, ac1P, and NSAC7 while PCRB measure benefits more from features NS1 and ac4R. It also shows that the features (e.g., NSAC5, ns2R, NS1, NSAC7) related to neighboring words of the negation signal play a critical role in recognizing both left and right boundaries. This may be due to the fact that neighboring words usually imply sentential information. For example, "*can not be*" indicates a passive clause while "*did not*" indicates an active clause. Table 3 also shows that the recognition of left boundaries is much easier than that of right boundaries. This may be due

[5] http://svmlight.joachims.org/

to the fact that 83.6% of negation signals have themselves as the left boundaries in the abstracts subcorpus.

Table 4 presents the performance on the abstracts subcorpus by performing 10-fold cross-validation. It shows that the additional features significantly improve the performance over the three measures ($\chi^2; p < 0.01$).

Feature	PCLB	PCRB	PCS
Baseline	84.29	87.82	74.05
+selected features	93.06	88.96	83.10

Table 4: Performance (%) of negation scope finding on the abstracts subcorpus using 10-fold cross-validation.

5.3 Experimental Results on Automatic Parse Trees

The GTB1.0 corpus contains 18,541 sentences in which 11,850 of them (63.91%) overlap with the sentences in the abstracts subcorpus[6]. In order to get automatic parse trees for the sentences in the abstracts subcorpus, we train the Berkeley parser with the remaining 6,691 sentences in GTB1.0. The Berkeley parser trained on 6,691 sentences achieves the performance of 85.22 in F1-measure on the other sentences in GTB1.0. For both the full papers and clinical reports subcorpora, we get their automatic parse trees by using two Berkeley parsers: one trained on 6,691 sentences in GBT1.0, and the other trained on all the sentences in GTB1.0.

To test the performance on automatic parse trees, we employ two different configurations. First, we train the argument identification classifier on the abstracts subcorpus using *automatic parse trees* produced by Berkeley parser trained on 6,691 sentences. The experimental results are presented in the rows of *autoparse(t&t)* in Table 5 and Table 6. Then, we train the argument identification classifier on the abstracts subcorpus using *golden parse trees*. The experimental results are presented in the rows of *autoparse(test)* in Table 5 and Table 6.

We also report an oracle performance to explore the best possible performance of our system by assuming that our negation scope finder can always correctly determine whether a candidate is an argument or not. That is, if an argument candidate is outside or cross-brackets with the golden negation scope, then it is a non-argument. The oracle performance is presented in the rows of *oracle* in Table 5 and Table 6.

Table 5 and Table 6 show that:

1) Automatic syntactic parsing lowers the performance of negation scope finding on the abstracts subcorpus in all three measures (e.g. from 83.10 to 81.84 in PCS). As expected, the parser trained on the whole GTB1.0 corpus works better than that trained on 6,691 sentences (e.g. 64.02 Vs. 62.70, and 89.79 Vs. 85.21 in PCS measure on the full papers and the clinical reports subcorpora, respectively). However, the performance decrease shows that negation scope finding is not as sensitive to automatic syntactic parsing as common shallow semantic parsing, whose performance might decrease by about ~10 in F1-measure (Toutanova et al., 2005). This indicates that negation scope finding via shallow semantic parsing is robust to some variations in the parse trees.
2) *autoparse(test)* consistently outperforms *autoparse(t&t)* on both the abstracts and the full papers subcorpora. However, it is surprising to find that *autoparse(t&t)* achieves better performance on the clinical reports subcorpus than *autoparse(test)*. This may be due to the special characteristics of the clinical reports subcorpus, which mainly consists of much shorter sentences with 6.6 words per sentence on average, and better adaptation of the argument identification classifier to the variations in the automatic parse trees.
3) The performance on all three subcorpora indicates that the recognition of right boundary is much harder than that of left boundary. This may be due to the longer right boundary on an average. Our statistics shows that the average left/right boundaries are 1.1/6.9, 0.1/3.7, and 1.2/6.5 words on the abstracts, the full papers and the clinical reports subcorpora, respectively.
4) The oracle performance is less sensitive to automatic syntactic parsing. In addition, given the performance gap between the performance of our negation scope finder and the oracle performance, there is still much room for further performance improvement.

[6] There are a few cases where two sentences in the abstracts subcorpus map into one sentence in GTB.

	Abstracts			Papers			Clinical		
	PCLB	PCRB	PCS	PCLB	PCRB	PCS	PCLB	PCRB	PCS
autoparse(t&t)	91.97	87.82	80.88	85.45	67.20	59.26	97.48	88.30	85.89
autoparse(test)	92.71	88.33	81.84	87.57	68.78	62.70	97.48	87.73	85.21
oracle	99.72	94.59	94.37	98.94	84.13	83.33	99.89	98.39	98.39

Table 5: Performance (%) of negation scope finding on the three subcorpora by using automatic parser trained with 6,691 sentences in GTB1.0.

	Papers			Clinical		
	PCLB	PCRB	PCS	PCLB	PCRB	PCS
autoparse(t&t)	85.98	67.99	60.32	97.48	92.66	90.48
autoparse(test)	87.83	70.11	64.02	97.36	92.20	89.79
oracle	98.94	83.86	83.07	99.77	97.94	97.82

Table 6: Performance (%) of negation scope finding on the two subcorpora by using automatic parser trained with all the sentences in GTB1.0.

Method	Abstracts	Papers	Clinical
M et al. (2008)	57.33	n/a	n/a
M & D (2009)	73.36	50.26	87.27
Our baseline	73.42	53.70	88.42
Our final system	81.84	64.02	89.79

Table 7: Performance comparison over the PCS measure (%) of our system with other state-of-the-art ones.

Table 7 compares our performance in PCS measure with related work. It shows that even our baseline system with four basic features as presented in Table 1 performs better than Morante et al. (2008) and Morante and Daelemans(2009). This indicates the appropriateness of our simplified shallow semantic parsing approach and the effectiveness of structured syntactic information on negation scope finding. It also shows that our final system significantly outperforms the state-of-the-art ones using a chunking approach, especially on the abstracts and full papers subcorpora. However, the improvement on the clinical reports subcorpus is less apparent, partly due to the fact that the sentences in this subcorpus are much simpler (with average length of 6.6 words per sentence) and thus a chunking approach can achieve high performance. Following are two typical sentences from the clinical reports subcorpus, where the negation scope covers the whole sentence (except the period punctuation). Such sentences account for 57% of negation sentences in the clinical reports subcorpus.

(1) No evidence of focal pneumonia .

(2) No findings to account for symptoms .

6 Conclusion

In this paper we have presented a simplified shallow semantic parsing approach to negation scope finding by formulating it as a shallow semantic parsing problem, which has been extensively studied in the past few years. In particular, we regard the negation signal as the predicate while mapping the negation scope into several constituents which are deemed as arguments of the negation signal. Evaluation on the Bioscope corpus shows the appropriateness of our shallow semantic parsing approach and that structured syntactic information plays a critical role in capturing the domination relationship between a negation signal and its negation scope. It also shows that our parsing approach much outperforms the state-of-the-art chunking ones. To our best knowledge, this is the first research on exploring negation scope finding via shallow semantic parsing.

Future research will focus on joint learning of negation signal and its negation scope findings. Although Morante and Daelemans (2009) reported the performance of 95.8%-98.7% on negation signal finding, it lowers the performance of negation scope finding by about 7.29%-16.52% in PCS measure.

Acknowledgments

This research was supported by Projects 60683150, 60970056, and 90920004 under the National Natural Science Foundation of China, Project 20093201110006 under the Specialized Research Fund for the Doctoral Program of Higher Education of China.

References

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *Proceedings of CoNLL 2005*.

Wendy W. Chapman, Will Bridewell, Paul Hanbury, Gregory F. Cooper, and Bruce G. Buchanan. 2001. A Simple Algorithm for Identifying Negated Findings and Diseases in Discharge Summaries. *Journal of Biomedical Informatics*, 34: 301-310.

Nigel Collier, Hyun Seok Park, Norihiro Ogata, et al. 1999. The GENIA project: corpus-based knowledge acquisition and information extraction from genome research papers. In *Proceedings of EACL 1999*.

Daniel Gildea and Martha Palmer. 2002. The Necessity of Parsing for Predicate Argument Recognition. In *Proceedings of ACL 2002*.

Ilya M. Goldin and Wendy W. Chapman. 2003. Learning to Detect Negation with 'Not' in Medical Texts. In *Proceedings of SIGIR 2003*.

Yang Huang and Henry Lowe. 2007. A Novel Hybrid Approach to Automated Negation Detection in Clinical Radiology Reports. *Journal of the American Medical Informatics Association*, 14(3): 304-311.

Zheng Ping Jiang and Hwee Tou Ng. 2006. Semantic Role Labeling of NomBank: A Maximum Entropy Approach. In *Proceedings of EMNLP 2006*.

Junhui Li, Guodong Zhou, Hai Zhao, Qiaoming Zhu, and Peide Qian. Improving Nominal SRL in Chinese Language with Verbal SRL Information and Automatic Predicate Recognition. In *Proceedings of EMNLP 2009*.

Roser Morante, Anthony Liekens, and Walter Daelemans. 2008. Learning the Scope of Negation in Biomedical Texts. In *Proceedings of EMNLP 2008*.

Roser Morante and Walter Daelemans. 2009. A Metalearning Approach to Processing the Scope of Negation. In *Proceedings of CoNLL 2009*.

Arzucan Özgür; Dragomir R. Radev. 2009. Detecting Speculations and their Scopes in Scientific Text. In *Proceedings of EMNLP 2009*.

Slav Petrov and Dan Klein. 2007. Improved Inference for Unlexicalized Parsing. In *Proceedings of NAACL 2007*.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2005. The Necessity of Syntactic Parsing for Semantic Role Labeling. In *Proceedings of IJCAI 2005*.

György Szarvas, Veronika Vincze, Richárd Farkas, and János Csirik. 2008. The BioScope corpus: annotation for negation, uncertainty and their scope in biomedical texts. In *Proceedings of BioNLP 2008*.

Yuka Tateisi, Akane Yakushiji, Tomoko Ohta, and Jun'ichi Tsujii. 2005. Syntax Annotation for the GENIA Corpus. In *Proceedings of IJCNLP 2005, Companion volume*.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2005. Joint Learning Improves Semantic Role Labeling. In *Proceedings of ACL 2005*.

Peter D. Turney. 2002. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In *Proceedings of ACL 2002*.

Veronika Vincze, György Szarvas, Richárd Farkas, György Móra, and János Csirik. 2008. The BioScope corpus: biomedical texts annotated for uncertainty, negation and their scopes. BMC Bioinformatics, 9(Suppl 11):S9.

Nianwen Xue and Martha Palmer. 2004. Calibrating Features for Semantic Role Labeling. In *Proceedings of EMNLP 2004*.

Nianwen Xue. 2008. Labeling Chinese Predicates with Semantic Roles. *Computational Linguistics*, 34(2):225-255.

Filtered Ranking for Bootstrapping in Event Extraction

Shasha Liao
Dept. of Computer Science
New York University
liaoss@cs.nyu.edu

Ralph Grishman
Dept. of Computer Science
New York University
grishman@cs.nyu.edu

Abstract

Several researchers have proposed semi-supervised learning methods for adapting event extraction systems to new event types. This paper investigates two kinds of bootstrapping methods used for event extraction: the document-centric and similarity-centric approaches, and proposes a filtered ranking method that combines the advantages of the two. We use a range of extraction tasks to compare the generality of this method to previous work. We analyze the results using two evaluation metrics and observe the effect of different training corpora. Experiments show that our new ranking method not only achieves higher performance on different evaluation metrics, but also is more stable across different bootstrapping corpora.

1 Introduction

The goal of event extraction is to identify instances of a class of events in text, along with the arguments of the event (the participants, place, and time). In this paper we shall focus on the sub-problem of identifying the events themselves.

Event extraction systems from the early and mid 90s relied primarily on hand-coded rules, which must be written anew for every task. Since then, supervised and semi-supervised methods have been developed in order to build systems for new scenarios more easily. Supervised methods can perform quite well with enough training data, but annotating sufficient data may require months of labor. Semi-supervised methods aim to reduce the annotated data required, ideally to a small set of seeds.

Most semi-supervised event extractors seek to learn sets of *patterns* consisting of a predicate and some lexical or semantic constraints on its arguments. The semi-supervised learning was based primarily on one of two assumptions: the document-centric approach, which assumes that relevant patterns should appear more frequently in relevant documents (Riloff 1996; Yangarber et al. 2000; Yangarber 2003; Surdeanu et al 2006); and the similarity-centric approach, which assumes that relevant patterns should have lexically related terms (Stevenson and Greenwood 2005, Greenwood and Stevenson 2006).

An effective semi-supervised extractor will have good performance over a range of extraction tasks and corpora. However, many of the learning procedures just cited have been tested on only one or two extraction tasks, so their generality is uncertain. To remedy this, we have tested learners based on both assumptions, targeting both a MUC (Message Understanding Conference) scenario and several ACE (Automatic Content Extraction) event types. We identify shortcomings of the prior bootstrapping methods, propose a more effective and stable ranking method, and consider the effect of different corpora and evaluation metrics.

2 Related Work

The basic assumption of the document-centric approach is that documents containing a large number of patterns already identified as relevant to a particular IE scenario are likely to contain further relevant patterns. Riloff (1996) initiated

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 680–688,
Beijing, August 2010

this approach and claimed that if a corpus can be divided into documents involving a certain event type and those not involving that type, patterns can be evaluated based on their frequency in relevant and irrelevant documents. Yangarber et al. (2000) incorporated Riloff's metric into a bootstrapping procedure, which started with several seed patterns but required no manual document classification or corpus annotation. The seed patterns were used to identify some relevant documents, and the top-ranked patterns (based on their distribution in relevant and irrelevant documents) were added to the seed set. This process was repeated, assigning a relevance score to each document based on the relevance of the patterns it contains and gradually growing the set of relevant patterns. This approach was further refined by Surdeanu et al. (2006), who used a co-training strategy in which two classifiers seek to classify documents as relevant to a particular scenario. Patwardhan and Riloff (2007) presented an information extraction system that find relevant regions of text and applies extraction patterns within those regions. They created a self-trained relevant sentence classifier to identify relevant regions, and use a semantic affinity measure to automatically learn domain-relevant extraction patterns. They also distinguish primary patterns from secondary patterns and apply the patterns selectively in the relevant regions.

Stevenson and Greenwood (2005) (henceforth 'S&G') suggested an alternative method for ranking the candidate patterns. Their approach relied on the assumption that useful patterns will have similar lexical items to the patterns that have already been accepted. They used WordNet to calculate word similarity. They chose to represent each pattern as a vector consisting of the lexical items and used a version of the cosine metric to determine the similarity between pairs of patterns. Later, Greenwood and Stevenson (2006) introduced a structural similarity measure that could be applied to extraction patterns consisting of linked dependency chains.

3 Ranking Methods in Bootstrapping

Most semi-supervised event extraction systems are based on patterns with variables which have semantic type constraints. A simple example is "*organization* appoints *person* as *position*"; if this pattern matches a passage in a test document, a hiring event will be instantiated with the items matching the variables being the arguments of the event. So training an event extractor becomes primarily a task of acquiring these patterns. In a semi-supervised setting, this involves ranking candidate patterns and accepting the top-ranked patterns at each iteration. Our goal was to create a more robust learner through improved pattern ranking.

3.1 Problems of Document-centric Bootstrapping

Document-centric bootstrapping tries to find patterns with high frequency in relevant documents and low frequency in irrelevant documents. The assumption is that descriptions of the same event or the same type of event may occur multiple times in a document, and so a document containing a relevant pattern is more likely to contain more such patterns. This approach may end up extracting patterns for related events; for example, *start-position* often comes with *end-position* events. This effect may be salutary if the extraction scenario includes these related events (as in MUC-6), but will pose a problem if the goal is to extract individual event types. Also, because an extra corpus for bootstrapping is needed, different corpora might perform quite differently (see Figure 2).

3.2 Problems of Similarity-centric Bootstrapping

Similarity-centric bootstrapping tries to find patterns with high lexical similarities. The most crucial issue is how to evaluate the similarity of two patterns, which is based on the similarity of two words. In this strategy, no extra corpus is needed, which eliminates the effort to find a good bootstrapping corpus, but a semantic dictionary that can provide word similarity is required. S&G used WordNet[1] to provide word similarity information. However, in the similarity-centric approach, lexical polysemy can lead the bootstrapping down false paths. For example, for *start-position (hire)* events, "*name*" and "*charge*" are in the same *Synset* as *appoint*, but including these words is quite dangerous because they contain other common senses

[1] http://wordnet.princeton.edu/

unrelated to *start-position* events. For *die* events, we might have words like "*go*" and "*pass*", which are also used in very specific contexts when they refer to "*die*". If similarity-centric ranking extracts patterns including these words, performance will deteriorate very quickly, because most of the time, these words do not predicate the proper event, and more and more wrong patterns will be extracted.

3.3 Our Approach

We propose a new ranking method, which constrains the document-centric and similarity-centric assumptions, and makes a more restricted assumption: patterns that appear in relevant documents *and* are lexically similar are most likely to be relevant. This method limits the effect of ambiguous patterns by narrowing the search to relevant documents, and limits irrelevant patterns in relevant documents by word similarity restriction. For example, although "charge" has high word similarity to "appoint", its document relevance score is very low, and we will not include this word in bootstrapping starting from "appoint".

Many different combinations are possible; we propose one that uses the word similarity as a filter. The document relevance score is first applied to rank the patterns in relevant documents, then the patterns with lexical similarity scores below a similarity threshold will be removed from the ranking; only patterns above threshold will be added to the seeds. However, if in the current iteration, no pattern meets the threshold, the threshold will be lowered until new patterns can be found. We call this ranking method *filtered ranking*[2]:

$$Filter(p) = \begin{cases} Yangarber(p) & Stevenson(p) >= t \\ 0 & otherwise \end{cases}$$

where t is the threshold, which is initialized to 0.9 in our experiments.

4 System Description

Our approach is similar to that for document-centric bootstrapping, but the ranking function is changed to incorporate lexical similarity information. For our experiments bootstrapping was terminated after a fixed number of iterations; in practice, we would monitor performance on a held-out (dev-test) sample and stop when it declines for k iterations.

4.1 Pre-processing

Instead of limiting ourselves to surface syntactic relations, we want to get more general and meaningful patterns. To this end, we used semantic role labeling (Gildea and Jurafsky, 2002) to generate the logical grammatical and predicate-argument representation automatically from a parse tree (Meyers et al. 2009). The output of the semantic labeling is the dependency representation of the text, where each sentence is a graph consisting of nodes (corresponding to words) and arcs. Each arc captures up to three relations between two words: (1) a SURFACE relation, the relation between a predicate and an argument in the parse tree of a sentence; (2) a LOGIC1 (grammatical logical) relation which regularizes for lexical and syntactic phenomena like passive, relative clauses, and deleted subjects; and (3) a LOGIC2 (predicate-argument) relation corresponding to relations in PropBank (Palmer et al. 2005) and NomBank

In constructing extraction patterns from this graph, we take each dependency link along with its predicate-argument role; if that role is null, we use its logical grammatical role, and finally, its surface role. For example, for the sentence:

```
John is hit by Tom's brother.
```

we generate the patterns:

<Arg1 hit John>
<Arg0 hit brother>
<T-pos brother Tom>

where the first two represent LOGIC2 relations and the third a SURFACE relation. To reduce data sparseness, all inflected words are changed to their root form (e.g. "attackers"→"attacker"), and all names are replaced by their ACE type (*person, organization, location, etc.*), so the first pattern would become

<Arg1 hit PERSON>

4.2 Document-based Ranking

The document-centric method employs a

[2] We also tried using the product of the document relevance score and word similarity score, and found the results to be quite similar. Due to space limitations, we do not report these results here.

re-implementation of the procedure described in (Yangarber et al. 2000), using the disjunctive voting scheme for document relevance. At each iteration i we compute a precision score $Prec^i(p)$ for each pattern p and a relevance score $Rel^i(d)$ for each document d. Initially the seed patterns have precision 1 and all other patterns precision 0. These are updated by

$$\mathrm{Re}\,l^i(d) = 1 - \prod_{p \in K(d)} (1 - \mathrm{Pr}\,ec^i(p))$$

where $K(d)$ is the set of accepted patterns that match document d, and

$$\mathrm{Pr}\,ec^{i+1}(p) = \frac{1}{|H(p)|} \bullet \sum_{d \in H(p)} \mathrm{Re}\,l^i(d)$$

where $H(p)$ is the set of documents matching pattern p. Patterns are then ranked by

$$RankFun_{Yangarber}(p) = \frac{Sup(p)}{|H(p)|} * \log Sup(p)$$

where $$Sup(p) = \sum_{d \in H(p)} \mathrm{Re}\,l(d)$$

(a generalization of Yangarber's metric), and the top-ranked candidates are added to the set of accepted patterns.

4.3 Pattern Similarity

For two words, there are several ways to measure their similarity using WordNet, which can be roughly divided into two categories: distance-based, including Leacock and Chodorow (1998), Wu and Palmer (1994); and information content based, including Resnik (1995), Lin (1998), and Jiang and Conrath (1997). We follow S&G (2005)'s method and use the semantic similarity of concepts based on Information Content (IC).

Every pattern consists of a predicate and a constraint ("argument") on its local syntactic context, and so the similarity of two patterns depends on the similarity of the predicates and the similarity of the arguments. We modified S&G's structural similarity measure to reflect some differences in pattern structure: first, S&G only focus on patterns headed by verbs, while we include verbs, nouns and adjectives; second, they only record the subject and object to a verb, while we record all argument relations; third, our patterns only contain a predicate and a single constraint (argument), while their pattern might contain two arguments, subject and object. With two arguments, many more patterns are possible and the vector similarity calculation over all patterns in a large corpus becomes very time consuming.

We do not limit ourselves to verb patterns because nouns and (occasionally) adjectives can also represent an event. For example, "Stevenson's promotion is a signal …" expresses a *start-position* event. Moreover, in our pattern, we assume that the predicate is more important than constraint, because it is the root (head) of the pattern in the semantic graph structure, and place different weights on predicate and constraint. Finally, the similarity of two patterns p_1 and p_2 is computed as follows:

$$\begin{aligned} &Sim(p_1, p_2) \\ &\quad = \alpha * Sim(f_1, f_2) + \beta * Sim(r_1, r_2) * Sim(a_1, a_2) \end{aligned}$$

where $\alpha+\beta=1$, f represents a predicate, r represent a role, and a represent an argument. In our experiment, α is set to 0.6 and β is set to 0.4. The role similarity is 1 for identical roles and for roles which generally correspond at the syntactic and predicate-argument level (arg0 ↔ subj; arg1 ↔ obj); selected other role pairs are assigned a small positive similarity (0.1 or 0.2), and others 0.

As with the document-centric method, bootstrapping begins by accepting a set of seed patterns. At each iteration, the procedure computes the similarity between all patterns in the training corpus and the currently accepted patterns and accepts the most similar pattern(s). In S&G's experiments the evaluation corpus also served as the training corpus.

5 Experiments

There have been two types of event extraction tasks. One involved several 'elementary' event types, such as "attack", "die", "injure" etc.; for example, the ACE 2005 evaluation[3] used a set of 33 event types and subtypes. The other type involved a *scenario* – a set of related events, like "attacks and the damage, injury, and death they cause", or "arrest, trial, sentencing etc.". The

[3]See http://projects.ldc.upenn.edu/ace/docs/English-Events-Guidelines_v5.4.3.pdf for a description of this task.

MUC evaluations included two scenarios that have been the subject of considerable research on learning methods: *terrorist incidents* (MUC-3/4) and *executive succession* (MUC-6).

We conducted experiments on the MUC-6 task to make a comparison to previous work. We also did experiments on ACE 2005 data, because it provides many distinct event types; we conducted experiments on three disparate event types: *attack*, *die*, and *start-position*. Note that MUC-6 identifies a scenario while ACE identifies specific event types, and types which are in the same MUC scenario might represent different ACE events. For example, the *executive succession* scenario (MUC-6) includes the *start-position* and *end-position* events in ACE.

5.1 Data Description

There are four corpora used in the experiments:

MUC-6 corpora

- **Bootstrapping:** pre-selected data from the Reuters corpus (Rose et al. 2002) from 1996 and 1997, including 3000 related documents and 3000 randomly chosen unrelated documents
- **Evaluation:** MUC-6 annotated data, including 200 documents (official training and test). We were guided by the MUC-6 key file in annotating every document and sentence as relevant or irrelevant.

ACE corpora

- **Bootstrapping:** untagged data from the Gigaword corpus from January 2006, including 14,171 English newswire articles from Agence France-Presse (AFP).
- **Evaluation:** ACE 2005 annotated (training) data, including 589 documents

5.2 Parameters used in Experiments

In our bootstrapping process, we only extract patterns appearing more than 2 times in the corpus, and the similarity filter threshold is originally set to 0.9. If no patterns are found, it is reduced by 0.1 until new patterns are found.

In each iteration, the top 3 patterns in the ranking function will be added to the seeds.

For the similarity-centric method, only patterns appearing more than 2 times and in less than 30% of the documents will be extracted, which is the same as S&G's approach.

5.3 MUC-6 Experiments

Our overall goal was to demonstrate that filtered ranking was in all cases competitive with and in at least some cases clearly superior to the earlier methods, over a range of extraction tasks and bootstrapping corpora. We began with the MUC-6 task, where the efficacy of the earlier methods had already been demonstrated.

< Arg0 resign Person >
< Arg1 appoint Person >
< Arg0 appoint Org_commercial>
<Arg1 succeed Person >

Table 1. Seeds for MUC-6 evaluation

For MUC-6 evaluation, we follow S&G's approach and assess extraction patterns by their ability to identify event-relevant sentences.[4] The system treats a sentence as relevant if it matches an extraction pattern. Bootstrapping starts from four seeds which yield 80% precision and 24% recall for sentence filtering.

To compare with previous work, we tested the filtered ranking method on two corpora: the first is the Reuters corpus used in S&G's recreation of Yangarber's experiment (Filter1), to compare with their results for the document-centric method; the second uses the test corpus as S&G did (Filter2), to compare with their results for the similarity-centric method. We compare methods based on peak F score; in practice, this would mean controlling the bootstrapping using a held-out test sample.

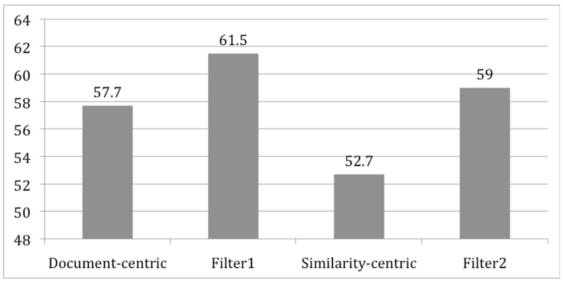


Figure 1. F score for different ranking methods on MUC-6 evaluation

Figure 1 showed that the filtered ranking

[4] We also tried the document filtering evaluation introduced by Yangarber but, as S&G observed, this metric is too insensitive because over 50% of the documents in the MUC-6 test set are relevant.

methods edge out both document and similarity-centric methods. Our scores are comparable to S&G's, although they report somewhat better performance for similarity-centric than for document-centric (55 vs. 51) whereas document-centric did better for us. This difference may reflect differences in pattern generation (discussed above) and possibly differences in the specific corpora used.

However, document-centric bootstrapping needs an extra corpus for bootstrapping; S&G used a pre-selected corpus that contains approximately same number of relevant and irrelevant documents[5]. We wanted to check if such a corpus is essential for the document-centric method, and if the need for pre-selection can be reduced through filtered ranking. Thus, we set up another experiment to see if the document-centric method is stable or sensitive to different corpora. We used two additional corpora for MUC-6 evaluation: one is a subset of the Wall Street Journal (WSJ) 1991 corpus, which contains 18,734 untagged documents; the other is the Gigaword AFP corpus described in section 5.1. Both corpora are much larger than the Reuters corpus, and while we do not have precise information about relevant document density, the WSJ contains quite a few *start-position* events because it is primarily business news; the Gigaword corpus (AFP newswire) has fewer *start-position* events because it contains a wider variety of news.

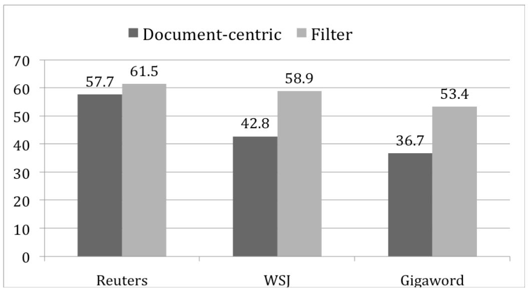


Figure 2. Document-centric and Filtered ranking results on different corpora for MUC-6

Figure 2 showed that the document-centric method performs quite differently on different corpora, which indicates that a pre-selected corpus plays an important role in document-centric ranking. It suggests that the percentage of relevant documents may be more important than the overall corpus size. The figure also shows that filtered ranking is much more stable across different corpora. Richer corpora still have better peak performance, but the difference is not quite as great; also, peak performance on a given corpus is consistently better than the document-centric method.

From the above experiments, we conclude that our filtering method is better in two aspects: first, bootstrapping on the same corpus performs better than either document or similarity-centric methods; second, if we can not get a corpus with an assured high density of relevant documents, it is safer to use filtered ranking because it is more stable across different corpora.

5.4 ACE2005 Experiments

The ACE2005 corpus includes annotations for 33 different event types and subtypes, offering us an opportunity to assess the generality of our methods across disparate event types. We selected 3 event types to report on here:

- **Die:** "occurs whenever the life of a PERSON Entity ends. It can be accidental, intentional or self-inflicted." This event appears 535 times in the corpus.
- **Attack:** "is defined as a violent physical act causing harm or damage." Attack events include a variety of sub-events like "person attack person", "country invade country", and "weapons attack locations". This event type appears 1120 times.
- **Start-Position:** "occurs whenever a PERSON Entity begins working for (or changes offices within) an ORGANIZATION or GPE. This includes government officials starting their terms, whether elected or appointed". It appears 116 times in the corpus.

We choose these three event types because they reflect the diversity of events ACE annotated: *die* events appear frequently in the ACE corpus and its definition is very clear; *attack* events also appear frequently, but its definition is rather complicated and contains several different sub-events; *start-position*'s definition is clear, but it is relatively infrequent in the corpus.

Based on the observations from the MUC-6 corpus, we eschewed corpus pre-selection for

[5] The pre-selection of relevant and irrelevant documents is based on document meta-data provided as part of the Reuters Corpus Volume I (Rose et al., 2002).

two reasons: first, building a different corpus for training each event type is an extra burden in developing a system for handling multiple events; second, we want to demonstrate that filtered ranking would work without pre-selection, while the document-centric method does not. As a result, we used the Gigaword AFP corpus for all event types.

In the ACE 2005 corpus, for every event, the annotators recorded a trigger, which is the main word that most clearly expresses an event occurrence. This added information allowed us to conduct dual evaluations: one based on sentence relevance - following S&G - presented in section 5.4.2, and one based on trigger identification, presented in section 5.4.3.

5.4.1 ACE2005 Supervised Model

To provide a benchmark for our semi-supervised learners, we built a very simple pattern-based supervised learning model. For training, for every pattern, we count how many times it contains an event trigger and how many times it does not. If more than 50% of the time it contains an event trigger, we treat it as a positive pattern.

For sentence level evaluation, if there is a positive pattern in a sentence, we tag this sentence as relevant; otherwise not. For word level evaluation, if the word is the predicator of a positive pattern, we tag it as a trigger; otherwise not[6].

We did a 5-fold cross-validation on the ACE 2005 data, report the average results and compare it to the semi-supervised learning method (see figure 3 & 4).

5.4.2 Sentence level ACE Event Evaluation[7]

Different event types have quite different performance (see figure 3): for the *die* event, the peak performance of all methods is quite good, and quite close to the supervised result; for the *attack* event, filtered ranking performs much better than both document and similarity-centric methods, but still worse than the supervised method; for *start-position* events, the semi-supervised method beats the supervised method. The reason might be as follows:

Die events appear frequently in ACE 2005, and most instances correspond to a small number of forms, so it is easy to find the correct patterns both from WordNet or related documents. As a result, filtered ranking provides no apparent benefit.

Attack is a more complicated event including several sub-events, which also have a lot of related events like *die* and *injure*. As a result, the document-centric method's performance goes down much faster, because patterns for related event types get drawn in; while the similarity-centric method performs worse than filtered ranking because some ambiguous words are introduced. For example, "hit" is an *attack* trigger, but words in the same Synset, such as "reach", "make", "attain", "gain" are quite dangerous because most of the time, these words do not refer to an attack event.

Start-position events do not appear frequently in ACE 2005, and supervised learning cannot achieve good performance because it can't collect enough training samples. The similarity-centric and Filter2 methods, which also depend on the ACE 2005 corpus, do not perform well either. Filter1 performs quite well because the Gigaword AFP corpus is quite large and contains more relevant documents, although the percentage is very small. This confirms our assumption that filtered ranking can achieve reasonable performance on a large unselected corpus, which is especially useful when the event is rare in the evaluation corpus.

<Arg1 kill Person>
<Arg1 slay Person>
<Arg1 death Person>

Table 2. Seeds for Ace 2005 *Die* evaluation

<Arg0 hire ORG>
<Arg1 hire Person>
<Arg1 appoint Person>
<Arg0 appoint ORG>

Table 3. Seeds for Ace 2005 *Start-Position* evaluation

[6]For word-level evaluation, we only consider trigger words with at least one semantic argument such as subject, object or a preposition; for that reason the performance is quite different from sentence level evaluation. We did the same for the word-level evaluation of semi-supervised learning.

[7] We do not list *Attack* seed patterns here as there are 34 patterns used.

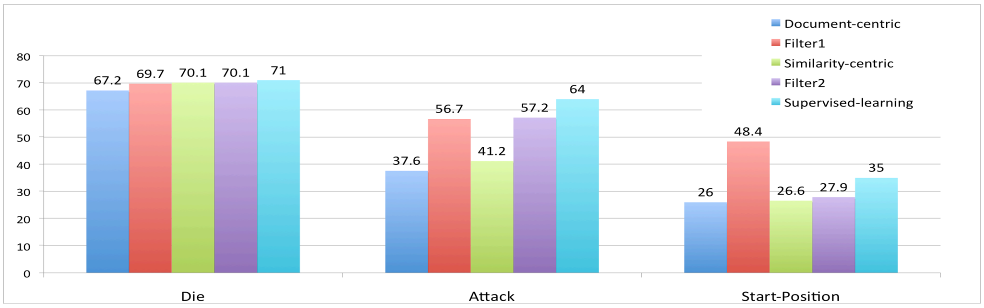


Figure 3. Performance on different ranking methods on ACE2005 sentence level evaluation

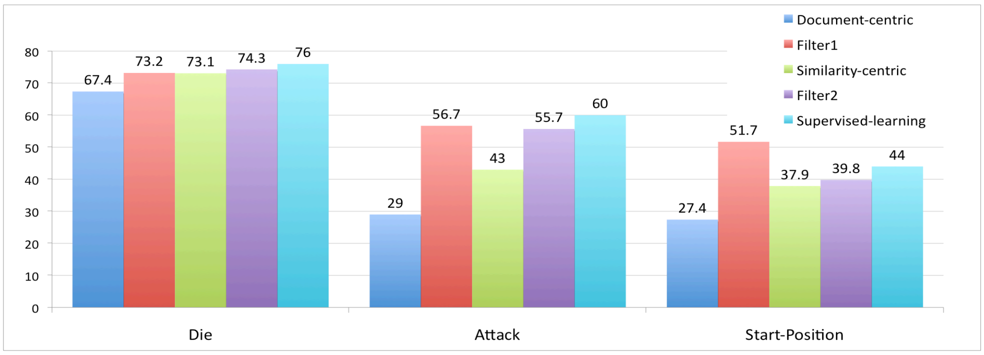


Figure 4. Performance on different ranking methods on ACE2005 word level evaluation

5.4.3 Word-level ACE Event Evaluation

Word-level evaluation is different from sentence-level evaluation because patterns which appear around an event but do not predicate an event are penalized in this evaluation. For example, the pattern <*Sbj chairman PERSON*>, which arises from a phrase like "*PERSON was the chairman of COMPANY*", appears much more in relevant *start-position* sentences than irrelevant sentences, and adding this pattern to the seeds will improve performance using the relevant-sentence metric. We would prefer a metric which discounted such patterns.

As noted above, ACE event annotations contain triggers, which are more specific event locators than a sentence, and we use this as the basis for a more specific evaluation. Extracted patterns are used to identify event triggers instead of identifying relevant sentences. For every word *w* in the ACE corpus, we extract all the patterns whose predicate is *w*. If the event extraction patterns include one of these patterns, we tag *w* as a trigger.

In word level evaluation, document-centric performs worse than the other methods. The reason is that some patterns appear often in the context of an event and are positive patterns for sentence level evaluation, but they do not actually predicate an event and are negative patterns in word level evaluation. In this situation, the document-centric method performs worse than the similarity-centric method, because it extracts many such patterns. For example, of the sentences which contain *die* events, 29% also contain *attack* events.

Thus in word level evaluation, filtered ranking continues to outperform either document- or similarity-centric methods, and its advantage over document-centric methods is accentuated.

6 Conclusions

In this paper, we propose a new ranking method in bootstrapping for event extraction and investigate the performance on different bootstrapping corpora with different ranking methods. This new method can block some irrelevant patterns coming from relevant documents, and, by preferring patterns from relevant documents, can eliminate some lexical ambiguity. Experiments show that this new ranking method performs better than previous ranking methods and is more stable across different corpora.

References

D. Gildea and D. Jurafsky. 2002. *Automatic Labeling of Semantic Roles.* Computational Linguistics, 28:245–288.

MA Greenwood, M. Stevenson. 2006. *Improving semi-supervised acquisition of relation extraction patterns.* Proceedings of the Workshop on Information Extraction Beyond the Document, pages 29–35.

Jay J. Jiang and David W. Conrath. 1997. *Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy.* In Proceedings of International Conference Research on Computational Linguistics (ROCLING X), Taiwan

C. Leacock and M. Chodorow. 1998. *Combining local context and WordNet similarity for word sense identification.* In C. Fellbaum, editor, WordNet: An electronic lexical database, pages 265–283. MIT Press.

D. Lin. 1998. *An information-theoretic definition of similarity.* In Proceedings of the International Conference on Machine Learning, Madison, August.

A. Meyers, M. Kosaka, N. Xue, H. Ji, A. Sun, S. Liao and W. Xu. 2009. Automatic Recognition of Logical Relations for English, Chinese and Japanese in the GLARF Framework. *In SEW-2009 (Semantic Evaluations Workshop) at NAACL HLT-2009*

MUC. 1995. Proceedings of the Sixth Message Understanding Conference (MUC-6), San Mateo, CA. Morgan Kaufmann.

Martha Palmer, Dan Gildea, and Paul Kingsbury, *The Proposition Bank: A Corpus Annotated with Semantic Roles,* Computational Linguistics, 31:1, 2005.

Siddharth Patwardhan, Satanjeev Banerjee, and Ted Pedersen. 2003. *Using measures of semantic relatedness for word sense disambiguation.* In Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, Mexico.

Patwardhan, S. and Riloff, E. 2007. *Effective Information Extraction with Semantic Affinity Patterns and Relevant Regions*. Proceedings of the 2007 Conference on Empirical Methods in Natural Language Processing (EMNLP-07)

T. Pedersen, S. Patwardhan, and J. Michelizzi. 2004. *WordNet::Similarity - Measuring the Relatedness of Concepts.* In Proceedings of the Nineteenth National Conference on Artificial Intelligence (Intelligent Systems Demonstrations), pages 1024-1025, San Jose, CA, July 2004.

P. Resnik. 1995. *Using information content to evaluate semantic similarity in a taxonomy.* In Proceedings of the 14th International Joint Conference on Artificial Intelligence, pages 448–453, Montreal, August.

Ellen Riloff. 1996. *Automatically Generating Extraction Patterns from Untagged Text.* In Proc. Thirteenth National Conference on Artificial Intelligence (AAAI-96), 1996, pp. 1044-1049.

T. Rose, M. Stevenson, and M. Whitehead. 2002. *The Reuters Corpus Volume 1 - from Yesterday's news to tomorrow's language resources.* In LREC-02, pages 827–832, La Palmas, Spain.

M. Stevenson and M. Greenwood. 2005. *A Semantic Approach to IE Pattern Induction.* Proceedings of ACL 2005.

Mihai Surdeanu, Jordi Turmo, and Alicia Ageno. 2006. *A Hybrid Approach for the Acquisition of Information Extraction Patterns.* Proceedings of the EACL 2006 Workshop on Adaptive Text Extraction and Mining (ATEM 2006)

Z. Wu and M. Palmer. 1994. *Verb semantics and lexical selection.* In 32nd Annual Meeting of the Association for Computational Linguistics, pages 133–138, Las Cruces, New Mexico.

Roman Yangarber; Ralph Grishman; Pasi Tapanainen; Silja Huttunen. 2000. *Automatic Acquisition of Domain Knowledge for Information Extraction.* Proc. COLING 2000.

Roman Yangarber. 2003. *Counter-Training in Discovery of Semantic Patterns.* Proceedings of ACL2003