Proceedings of

SSST-4

Fourth Workshop on

Syntax and Structure in Statistical Translation

Dekai Wu (editor)

COLING 2010 / SIGMT Workshop
23rd International Conference on Computational Linguistics
Beijing, China
28 August 2010

Produced by
Chinese Information Processing Society of China

To order the CD of Coling 2010 and its Workshop Proceedings, please 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

Introduction

The Fourth Workshop on Syntax and Structure in Statistical Translation (SSST-4) was held on 28 August 2010 following the Coling 2010 conference in Beijing. Like the first three SSST workshops in 2007, 2008, and 2009, it aimed to bring together researchers from different communities working in the rapidly growing field of statistical, tree-structured models of natural language translation.

We were honored to have Martin Kay deliver this year's invited keynote talk. This field is indebted to Martin Kay for not one but *two* of the classic cornerstone ideas that inspired bilingual tree-structured models for statistical machine translation: first, chart parsing, and second, parallel text alignment.

Tabular approaches to parsing, using dynamic programming and/or memoization, were heavily influenced by Kay's (1980) charts (or forests, packed forests, well-formed substring tables, or WFSTs). Today's biparsing models—the bilingual generalizations of this influential work—lie at the heart of numerous alignment and training algorithms for inversion transduction grammars or ITGs—including all syntax-directed transduction grammars or SDTGs (or synchronous CFGs) of binary rank or ternary rank, such as those learned by hierarchical phrase-based translation approaches.

At the same time, Kay and Röscheisen's (1988) seminal work on alignment of parallel texts led the way in statistical machine translation's basic paradigm of integrating the simultaneous learning of translation lexicons with aligning parallel texts. Today's biparsing models generalize this by simultaneously learning tree structures as well. Once again, cross-pollination of ideas across different areas and disciplines, empirical and theoretical, has provided mutual inspiration.

We selected 15 papers for this year's workshop. Studies emphasizing formal and algorithmic aspects include a method for intersecting synchronous/transduction grammars (S/TGs) with finite-state transducers (Dymetman and Cancedda), and a comparison of linear transduction grammars (LTGs) with ITGs (Saers, Nivre and Wu). Experiments on using syntactic features and constraints within flat phrase-based translation models include studies by Jiang, Du and Way; by Cao, Finch and Sumita; by Kolachina, Venkatapathy, Bangalore, Kolachina and PVS; and by Zhechev and van Genabith. Dependency constraints are also used to improve HMM word alignments for both flat phrase-based as well as S/TG based translation models (Ma and Way). Extensions to the features and parameterizations in two S/TG based translation models, as well as methods for merging models, are studied by Zollman and Vogel. The potential of incorporating LFG-style deep syntax within S/TGs is explored by Graham and van Genabith. A tree transduction based approach is presented by Khalilov and Sima'an. Meanwhile, Lo and Wu empirically compare n-gram, syntactic, and semantic structure based MT evaluation approaches. An encouraging trend is an uptick in work on low-resource language pairs and from underrepresented regions, including English-Persian (Mohaghegh, Sarrafzadeh and Moir), Manipuri-English (Singh and Bandyopadhyay), Tunisia (Khemakhem, Jamoussi and Ben hamadou), and English-Hindi (Venkatapathy, Sangal, Joshi, and Gali).

Once again this year, thanks are due to our authors and our Program Committee for making the SSST workshop another success.

Acknowledgements

This work was supported in part by the Defense Advanced Research Projects Agency (DARPA) under GALE Contract Nos. HR0011-06-C-0022, subcontract BBN Technologies and HR0011-06-C-0023, subcontract SRI International. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency.

Organizers:

Dekai WU, Hong Kong University of Science and Technology (HKUST), Hong Kong

Program Committee:

Srinivas BANGALORE, AT&T Research, USA
Marine CARPUAT, Hong Kong University of Science and Technology (HKUST), Hong Kong
David CHIANG, USC Information Sciences Institute, USA
Pascale FUNG, Hong Kong University of Science and Technology (HKUST), Hong Kong
Daniel GILDEA, University of Rochester, USA
Dan KLEIN, University of California at Berkeley, USA
Kevin KNIGHT, USC Information Sciences Institute, USA
Jonas KUHN, University of Potsdam, Germany
Yang LIU, Institute of Computing Technology, Chinese Academy of Sciences, China
Yanjun MA, Dublin City University, Ireland
Daniel MARCU, USC Information Sciences Institute, USA
Yuji MATSUMOTO, Nara Institute of Science and Technology, Japan
Hermann NEY, RWTH Aachen, Germany
Owen RAMBOW, Columbia University, USA
Philip RESNIK, University of Maryland, USA
Stefan RIEZLER, Google Inc., USA
Libin SHEN, BBN Technologies, USA
Christoph TILLMANN, IBM T. J. Watson Research Center, USA
Stephan VOGEL, Carnegie Mellon University, USA
Taro WATANABE, NTT Communication Science Laboratories, Japan
Andy WAY, Dublin City University, Ireland
Yuk-Wah WONG, Google Inc., USA
Richard ZENS, Google Inc., USA
Chengqing ZONG, Institute of Automation, Chinese Academy of Sciences, China

Table of Contents

Conference Program

Saturday, August 28, 2010

9:00–10:40 *Open Tutorial: Tree-Structured and Syntactic SMT*
Dekai Wu

10:40–11:05 Coffee break

11:05–12:05 *Invited Keynote*
Martin Kay

12:05–12:25 *Intersecting Hierarchical and Phrase-Based Models of Translation: Formal Aspects and Algorithms*
Marc Dymetman and Nicola Cancedda

12:25–12:55 *A Systematic Comparison between Inversion Transduction Grammar and Linear Transduction Grammar for Word Alignment*
Markus Saers, Joakim Nivre and Dekai Wu

12:55–14:00 Lunch break

14:00–14:20 *Source-side Syntactic Reordering Patterns with Functional Words for Improved Phrase-based SMT*
Jie Jiang, Jinhua Du and Andy Way

14:20–14:40 *Syntactic Constraints on Phrase Extraction for Phrase-Based Machine Translation*
Hailong Cao, Andrew Finch and Eiichiro Sumita

14:40–15:00 *Phrase Based Decoding using a Discriminative Model*
Prasanth Kolachina, Sriram Venkatapathy, Srinivas Bangalore, Sudheer Kolachina and Avinesh PVS

15:00–15:20 *Seeding Statistical Machine Translation with Translation Memory Output through Tree-Based Structural Alignment*
Ventsislav Zhechev and Josef van Genabith

15:20–15:40 *Semantic vs. Syntactic vs. N-gram Structure for Machine Translation Evaluation*
Chi-kiu Lo and Dekai Wu

15:40–16:05 Posters / Coffee break

Arabic morpho-syntactic feature disambiguation in a translation context
Ines Turki Khemakhem, Salma Jamoussi and Abdelmajid Ben hamadou

Saturday, August 28, 2010 (continued)

A Discriminative Approach for Dependency Based Statistical Machine Translation
Sriram Venkatapathy, Rajeev Sangal, Aravind Joshi and Karthik Gali

Improved Language Modeling for English-Persian Statistical Machine Translation
Mahsa Mohaghegh, Abdolhossein Sarrafzadeh and Tom Moir

Manipuri-English Bidirectional Statistical Machine Translation Systems using Morphology and Dependency Relations
Thoudam Doren Singh and Sivaji Bandyopadhyay

16:05–16:25 *A Discriminative Syntactic Model for Source Permutation via Tree Transduction*
Maxim Khalilov and Khalil Sima'an

16:25–16:45 *HMM Word-to-Phrase Alignment with Dependency Constraints*
Yanjun Ma and Andy Way

16:45–17:05 *New Parameterizations and Features for PSCFG-Based Machine Translation*
Andreas Zollmann and Stephan Vogel

17:05–17:25 *Deep Syntax Language Models and Statistical Machine Translation*
Yvette Graham and Josef van Genabith

17:25–17:40 Discussion

Intersecting Hierarchical and Phrase-Based Models of Translation: Formal Aspects and Algorithms

Marc Dymetman **Nicola Cancedda**
Xerox Research Centre Europe
{marc.dymetman,nicola.cancedda}@xrce.xerox.com

Abstract

We address the problem of constructing hybrid translation systems by intersecting a Hiero-style hierarchical system with a phrase-based system and present formal techniques for doing so. We model the phrase-based component by introducing a variant of weighted finite-state automata, called σ-automata, provide a self-contained description of a general algorithm for intersecting weighted synchronous context-free grammars with finite-state automata, and extend these constructs to σ-automata. We end by briefly discussing complexity properties of the presented algorithms.

1 Introduction

Phrase-based (Och and Ney, 2004; Koehn et al., 2007) and Hierarchical (Hiero-style) (Chiang, 2007) models are two mainstream approaches for building Statistical Machine Translation systems, with different characteristics. While phrase-based systems allow a direct capture of correspondences between surface-level lexical patterns, but at the cost of a simplistic handling of re-ordering, hierarchical systems are better able to constrain re-ordering, especially for distant language pairs, but tend to produce sparser rules and often lag behind phrase-based systems for less distant language pairs. It might therefore make sense to capitalize on the complementary advantages of the two approaches by combining them in some way.

This paper attempts to lay out the formal prerequisites for doing so, by developing techniques for intersecting a hierarchical model and a phrase-based model. In order to do so, one first difficulty has to be overcome: while hierarchical systems are based on the mathematically well-understood formalism of weighted synchronous CFG's, phrase-based systems do not correspond to any classical formal model, although they are loosely connected to weighted finite state transducers, but crucially go beyond these by allowing phrase re-orderings.

One might try to address this issue by limiting *a priori* the amount of re-ordering, in the spirit of (Kumar and Byrne, 2005), which would allow to approximate a phrase-based model by a standard transducer, but this would introduce further issues. First, limiting the amount of reordering in the phrase-based model runs contrary to the underlying intuitions behind the intersection, namely that the hierarchical model should be mainly responsible for controlling re-ordering, and the phrase-based model mainly responsible for lexical choice. Second, the transducer resulting from the operation could be large. Third, even if we could represent the phrase-based model through a finite-state transducer, intersecting this transducer with the synchronous CFG would actually be intractable in the general case, as we indicate later.

We then take another route. For a fixed source sentence x, we show how to construct an automaton that represents all the (weighted) target sentences that can be produced by applying the phrase based model to x. However, this "σ-automaton" is non-standard in the sense that each transition is decorated with a set of source sentence tokens and that the only valid paths are

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 1–9,
COLING 2010, Beijing, August 2010.

those that do not traverse two sets containing the same token (in other words, valid paths cannot "consume" the same source token twice).

The reason we are interested in σ-automata is the following. First, it is known that intersecting a synchronous grammar simultaneously with the source sentence x and a (standard) target automaton results in another synchronous grammar; we provide a self-contained description of an algorithm for performing this intersection, in the general weighted case, and where x is generalized to an arbitrary source automaton. Second, we extend this algorithm to σ-automata. The resulting weighted synchronous grammar represents, as in Hiero, the "parse forest" (or "hypergraph") of all weighted derivations (that is of all translations) that can be built over x, but where the weights incorporate knowledge of the phrase-based component; it can therefore form the basis of a variety of dynamic programming or sampling algorithms (Chiang, 2007; Blunsom and Osborne, 2008), as is the case with standard Hiero-type representations. While in the worst case the intersected grammar can contain an exponential number of nonterminals, we argue that such combinatorial explosion will not happen in practice, and we also briefly indicate formal conditions under which it will not be *allowed* to happen.

2 Intersecting weighted synchronous CFG's with weighted automata

We assume that the notions of weighted finite-state automaton [W-FSA] and weighted synchronous grammar [W-SCFG] are known (for short descriptions see (Mohri et al., 1996) and (Chiang, 2006)), and we consider:

1. A W-SCFG G, with associated source grammar G_s (resp. target grammar G_t); the terminals of G_s (resp. G_t) vary over the source vocabulary V_s (resp. target vocabulary V_t).
2. A W-FSA A_s over the source vocabulary V_s, with initial state $s_\#$ and final state $s_\$$.
3. A W-FSA A_t over the target vocabulary V_t, with initial state $t_\#$ and final state $t_\$$.

The grammar G defines a weighted synchronous language L_G over (V_s, V_t), the automaton A_s a weighted language L_s over V_s, and the automaton A_t a weighted language L_t over V_t. We then define the intersection language L' between these three languages as the synchronous language denoted $L' = L_s \Cap L_G \Cap L_t$ over (V_s, V_t) such that, for any pair (x, y) of a source and a target sentence, the weight $L'(x, y)$ is defined by $L'(x, y) \equiv L_s(x) \cdot L_G(x, y) \cdot L_t(y)$, where $L_s(x), L_G(x, y), L_t(y)$ are the weights associated to each of the component languages.

It is natural to ask whether there exists a synchronous grammar G' generating the language L', which we will now show to be the case.[1] Our approach is inspired by the construction in (Bar-Hillel et al., 1961) for the intersection of a CFG and an FSA and the observation in (Lang, 1994) relating this construction to parse forests, and also partially from (Satta, 2008), although, by contrast to that work, our construction, (i) is done simultaneously rather than as the sequence of intersecting A_s with G, then the resulting grammar with A_t, (ii) handles weighted formalisms rather than non-weighted ones.

We will describe the construction of G' based on an example, from which the general construction follows easily. Consider a W-SCFG grammar G for translating between French and English, with initial nonterminal S, and containing among others the following rule:

$$\mathrm{N} \rightarrow \text{A manque à B / B misses A} : \theta, \qquad (1)$$

where the source and target right-hand sides are separated by a slash symbol, and where θ is a non-negative real weight (interpreted multiplicatively) associated with the rule.

Now let's consider the following "rule scheme":

$$\begin{aligned} {}^{t_0}_{s_0}\mathrm{N}^{t_3}_{s_4} \rightarrow {}^{t_2}_{s_0}\mathrm{A}^{t_3}_{s_1}\ {}_{s_1}\text{manque}_{s_2}\ {}_{s_2}\text{à}_{s_3}\ {}^{t_0}_{s_3}\mathrm{B}^{t_1}_{s_4} \quad / \\ {}^{t_0}_{s_3}\mathrm{B}^{t_1}_{s_4}\ {}^{t_1}\text{misses}^{t_2}\ {}^{t_2}_{s_0}\mathrm{A}^{t_3}_{s_1} \end{aligned} \qquad (2)$$

[1] We will actually only need the application of this result to the case where A_s is a "degenerate" automaton describing a single source sentence x, but the general construction is not harder to do than this special case and the resulting format for G' is well-suited to our needs below.

This scheme consists in an "indexed" version of the original rule, where the bottom indices s_i correspond to states of A_s ("source states"), and the top indices t_i to states of A_t ("target states"). The nonterminals are associated with two source and two target indices, and for the same nonterminal, these four indices have to match across the source and the target RHS's of the rule. As for the original terminals, they are replaced by "indexed terminals", where source (resp. target) terminals have two source (resp. target) indices. The source indices appear sequentially on the source RHS of the rule, in the pattern $s_0, s_1, s_1, s_2, s_2 \dots s_{m-1}, s_m$, with the nonterminal on the LHS receiving source indices s_0 and s_m, and similarly the target indices appear sequentially on the target RHS of the rule, in the pattern $t_0, t_1, t_1, t_2, t_2 \dots t_{n-1}, t_n$, with the nonterminal on the LHS receiving target indices t_0 and t_n. To clarify, the operation of associating indices to terminals and nonterminals can be decomposed into three steps:

$$
\begin{aligned}
{}_{s_0}\mathrm{N}_{s_4} &\rightarrow {}_{s_0}\mathrm{A}_{s_1}\ {}_{s_1}\text{manque}_{s_2}\ {}_{s_2}\text{à}\ {}_{s_3}\ {}_{s_3}\mathrm{B}_{s_4}\ /\\
&\qquad \text{B misses A}\\
{}^{t_0}\mathrm{N}^{t_3} &\rightarrow \text{A manque à B}\ /\\
&\qquad {}^{t_0}\mathrm{B}^{t_1}\ {}^{t_1}\text{misses}^{t_2}\ {}^{t_2}\mathrm{A}^{t_3}\\
{}^{t_0}_{s_0}\mathrm{N}^{t_3}_{s_4} &\rightarrow {}^{t_2}_{s_0}\mathrm{A}^{t_3}_{s_1}\ {}_{s_1}\text{manque}_{s_2}\ {}_{s_2}\text{à}\ {}_{s_3}\ {}^{t_0}_{s_3}\mathrm{B}^{t_1}_{s_4}\ /\\
&\qquad {}^{t_0}_{s_3}\mathrm{B}^{t_1}_{s_4}\ {}^{t_1}\text{misses}^{t_2}\ {}^{t_2}_{s_0}\mathrm{A}^{t_3}_{s_1}
\end{aligned}
$$

where the first two steps corresponds to handling the source and target indices separately, and the third step then assembles the indices in order to get the same four indices on the two copies of each RHS nonterminal. The rule scheme (2) now generates a family of rules, each of which corresponds to an *arbitrary* instantiation of the source and target indices to states of the source and target automata respectively. With every such rule instantiation, we associate a weight θ' which is defined as:

$$
\begin{aligned}
\theta' \equiv \theta \cdot & \prod_{{}_{s_i}\text{s-term}_{s_{i+1}}} \theta_{A_s}(s_i, \text{s-term}, s_{i+1})\\
\cdot & \prod_{{}^{t_j}\text{t-term}^{t_{j+1}}} \theta_{A_t}(t_j, \text{t-term}, t_{j+1}), \quad (3)
\end{aligned}
$$

where the first product is over the indexed source terminals ${}_{s_i}\text{s-term}_{s_{i+1}}$, the second product over the indexed target terminals ${}^{t_j}\text{t-term}^{t_{j+1}}$; $\theta_{A_s}(s_i, \text{s-term}, s_{i+1})$ is the weight of the transition $(s_i, \text{s-term}, s_{i+1})$ according to A_s, and similarly for $\theta_{A_t}(t_j, \text{t-term}, t_{j+1})$. In these products, it may happen that $\theta_{A_s}(s_i, \text{s-term}, s_{i+1})$ is null (and similarly for A_t), and in such a case, the corresponding rule instantiation is considered not to be realized. Let us consider the multiset of all the weighted rule instantiations for (1) computed in this way, and for each rule in the collection, let us "forget" the indices associated to the terminals. In this way, we obtain a collection of weighted synchronous rules over the vocabularies V_s and V_t, but where each nonterminal is now indexed by four states.[2]

When we apply this procedure to all the rules of the grammar G, we obtain a new weighted synchronous CFG G', with start symbol ${}^{t_\#}_{s_\#}S^{t_\$}_{s_\$}$, for which we have the following **Fact**, of which we omit the proof for lack of space.

Fact 1. *The synchronous language $L_{G'}$ associated with G' is equal to $L' = L_s \Cap L_G \Cap L_t$.*

The grammar G' that we have just constructed does fulfill the goal of representing the bilateral intersection that we were looking for, but it has a serious defect: most of its nonterminals are *improductive*, that is, can never produce a bi-sentence. If a rule refers to such an improductive nonterminal, it can be eliminated from the grammar. This is the analogue for a SCFG of the classical operation of *reduction* for CFG's; while, conceptually, we could start from G' and perform the reduction by *deleting* the many rules containing improductive nonterminals, it is equivalent but much more efficient to do the reverse, namely to incrementally *add* the productive nonterminals and rules of G' starting from an initially empty set of rules, and by proceeding bottom-up starting from the terminals. We do not detail this process, which is relatively

[2] It is possible that the multiset obtained by this simplifying operation contains duplicates of certain rules (possibly with different weights), due to the non-determinism of the automata: for instance, two sequences such as '${}_{s_1}\text{manque}_{s_2}\ {}_{s_2}\text{à}_{s_3}$' and '${}_{s_1}\text{manque}_{s'_2}\ {}_{s'_2}\text{à}_{s_3}$' become indistinguishable after the operation. Rather than producing multiple instances of rules in this way, one can "conflate" them together and add their weights.

straightforward.[3]

A note on intersecting SCFGs with transducers Another way to write $L_s \Cap L_G \Cap L_t$ is as the intersection $(L_s \times L_t) \cap L_G$. $(L_s \times L_t)$ can be seen as a rational language (language generated by a finite state transducer) of an especially simple form over $V_s \times V_t$. It is then natural to ask whether our previous construction can be generalized to the intersection of G with an arbitrary finite-state transducer. However, this is not the case. Deciding the emptiness problem for the intersection between two finite state transducers is already undecidable, by reduction to Post's Correspondence problem (Berstel, 1979, p. 90) and we have extended the proof of this fact to show that intersection between a synchronous CFG and a finite state transducer also has an undecidable emptiness problem (the proof relies on the fact that a finite state transducer can be simulated by a synchronous grammar). *A fortiori*, this intersection cannot be represented through an (effectively constructed) synchronous CFG.

3 Phrase-based models and σ-automata

3.1 σ-automata: definition

Let V_s be a source vocabulary, V_t a target vocabulary. Let $x = x_1, \ldots, x_M$ be a *fixed* sequence of words over a certain source vocabulary V_s. Let us denote by z a *token* in the sequence x, and by Z the set of the M tokens in x. A σ-automaton over x has the general form of a standard weighted automaton over the target vocabulary, but where the edges are also decorated with elements of $\mathcal{P}(Z)$, the powerset of Z (see Fig. 1). An edge in the σ-automaton between two states q and q' then carries a label of the form (α, β), where $\alpha \in \mathcal{P}(Z)$ and $\beta \in V_t$ (note that here we do not allow β to be the empty string ϵ). A path from the initial state of the automaton to its final state is defined to be *valid* iff each token of x appears in exactly one label of the path, but not necessarily in the same order as in x. As usual, the output associated with the path is the ordered sequence of target labels on that path, and the weight of the path is the product of the weights on its edges.

σ-automata and phrase-based translation A mainstream phrase-based translation system such as Moses (Koehn et al., 2007) can be accounted for in terms of σ-automata in the following way. To simplify exposition, we assume that the language model used is a bigram model, but any n-gram model can be accommodated. Then, *given a source sentence* x, decoding works by attempting to construct a sequence of phrase-pairs of the form $(\tilde{x}_1, \tilde{y}_1), ..., (\tilde{x}_k, \tilde{y}_k)$ such that each $\tilde{x}_i$ corresponds to a contiguous subsequence of *tokens* of x, the $\tilde{x}_i$'s do not overlap and completely cover x, but may appear in a different order than that of x; the output associated with the sequence is simply the concatenation of all the $\tilde{y}_i$'s in that sequence.[4] The weight associated with the sequence of phrase-pairs is then the product (when we work with probabilities rather than log-probabilities) of the weight of each $(\tilde{x}_{i+1}, \tilde{y}_{i+1})$ in the context of the previous $(\tilde{x}_i, \tilde{y}_i)$, which consists in the product of several elements: (i) the "out-of-context" weight of the phrase-pair $(\tilde{x}_{i+1}, \tilde{y}_{i+1})$ as determined by its features in the phrase table, (ii) the language model probability of finding $\tilde{y}_{i+1}$ following $\tilde{y}_i$,[5] (iii) the contextual weight of $(\tilde{x}_{i+1}, \tilde{y}_{i+1})$ relative to $(\tilde{x}_i, \tilde{y}_i)$ corresponding to the distorsion cost of "jumping" from the token sequence $\tilde{x}_i$ to the token sequence $\tilde{x}_{i+1}$ when these two sequences may not be consecutive in x.[6]

Such a model can be represented by a σ-automaton, where each phrase-pair $(\tilde{x}, \tilde{y})$ — for

[3] This bottom-up process is analogous to *chart-parsing*, but here we have decomposed the construction into first building a semantics-preserving grammar and then reducing it, which we think is formally neater.

[4] We assume here that the phrase-pairs $(\tilde{x}_i, \tilde{y}_i)$ are such that $\tilde{y}_i$ is not the empty string (this constraint could be removed by an adaptation of the ϵ-removal operation (Mohri, 2002) to σ-automata).

[5] This is where the bigram assumption is relevant: for a trigram model, we may need to encode in the automaton not only the immediately preceding phrase-pair, but also the previous one, and so on for higher-order models. An alternative is to keep the n-gram language model outside the σ-automaton and intersect it later with the grammar G' obtained in section 4, possibly using approximation techniques such as cube-pruning (Chiang, 2007).

[6] Any distorsion model — in particular "lexicalized reordering" — that only depends on comparing two consecutive phrase-pairs can be implemented in this way.

p-transducer

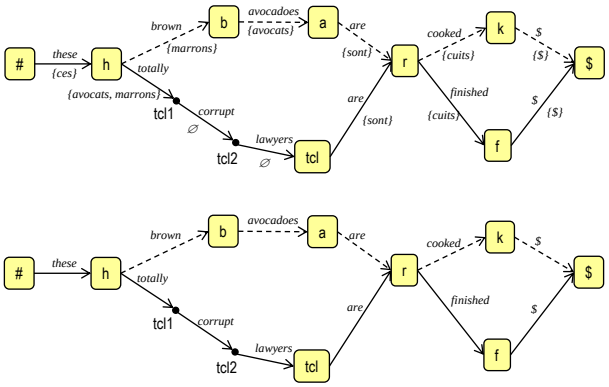


Figure 1: **On the top:** a σ-automaton with two valid paths shown. Each box denotes a state corresponding to a phrase pair, while states internal to a phrase pair (such as tcl1 and tcl2) are not boxed. Above each transition we have indicated the corresponding target word, and below it the corresponding set of source tokens. We use a terminal symbol \$ to denote the end of sentence both on the source and on the target. The solid path corresponds to the output *these totally corrupt lawyers are finished*, the dotted path to the output *these brown avocadoes are cooked*. Note that the source tokens are not necessarily consumed in the order given by the source, and that, for example, there exists a valid path generating *these are totally corrupt lawyers finished* and moving according to $h \rightarrow r \rightarrow tcl1 \rightarrow tcl2 \rightarrow tcl \rightarrow f$; Note, however, that this does not mean that if a biphrase such as `(marrons avocats, avocado chestnuts)` existed in the phrase table, it would be applicable to the source sentence here: because the source words in this biphrase would not match the order of the source tokens in the sentence, the biphrase would not be included in the σ-automaton at all. **On the bottom:** The target W-FSA automaton A_t associated with the σ-automaton, where we are ignoring the source tokens (but keeping the same weights).

$\tilde{x}$ a sequence of tokens in x and $(\tilde{x}, \tilde{y})$ an entry in the global phrase table — is identified with a state of the automaton and where the fact that the phrase-pair $(\tilde{x}', \tilde{y}') = ((x_1, ..., x_k), (y_1, ..., y_l))$ follows $(\tilde{x}, \tilde{y})$ in the decoding sequence is modeled by introducing l "internal" transitions with labels (σ, y_1), $(\emptyset, y_2)$, ..., $(\emptyset, y_l)$, where $\sigma = \{x_1, ..., x_k\}$, and where the first transition connects the state $(\tilde{x}, \tilde{y})$ to some unique "internal state" q_1, the second transition the state q_1 to some unique internal state q_2, and the last transition q_k to the state $(\tilde{x}', \tilde{y}')$.[7] Thus, a state $(\tilde{x}', \tilde{y}')$ essentially encodes the previous phrase-pair used during decoding, and it is easy to see that it is possible to account for the different weights associated with the phrase-based model by weights associated to the transitions of the σ-automaton.[8]

Example Let us consider the following French source sentence x: *ces avocats marrons sont cuits* (idiomatic expression for *these totally corrupt lawyers are finished*). Let's assume that the phrase table contains the following phrase pairs:

```
h: (ces, these)
a: (avocats, avocados)
b: (marrons, brown)
tcl: (avocats marrons,
            totally corrupt lawyers)
r: (sont, are)
k: (cuits, cooked)
f: (cuits, finished).
```

An illustration of the corresponding σ-automaton SA is shown at the top of Figure 1, with only a few transitions made explicit, and with no weights shown.[9]

[7] For simplicity, we have chosen to collect the set of all the source tokens $\{x_1, ..., x_k\}$ on the first transition, but we could distribute it on the l transitions arbitrarily (but keeping the subsets disjoint) without changing the semantics of what we do. This is because once we have entered one of the l internal transitions, we will always have to traverse the remaining internal transitions and collect the full set of source tokens.

[8] By creating states such as $((\tilde{x}, \tilde{y}), (\tilde{x}', \tilde{y}'))$ that encode the two previous phrase-pairs used during decoding, it is possible in principle to account for a trigram language model, and similarly for higher-order LMs. This is similar to implementing n-gram language models by automata whose states encode the $n-1$ words previously generated.

[9] Only two (valid) paths are shown. If we had shown the full σ-automaton, then the graph would have been "complete" in the sense that for any two box states B, B', we would have shown a connection $B \rightarrow B'_1 ... \rightarrow B'_{k-1} \rightarrow B'$, where the B'_i are internal states, and k is the length of the target side of the biphrase B'.

4 Intersecting a synchronous grammar with a σ-automaton

Intersection of a W-SCFG with a σ-automaton If SA is a σ-automaton over input x, with each valid path in SA we associate a weight in the same way as we do for a weighted automaton. For any target word sequence in V_t^* we can then associate the sum of the weights of all valid paths outputting that sequence. The weighted language $L_{SA,x}$ over V_t obtained in this way is called the language associated with SA. Let

V_s, V_t, and let us denote

language over V_s, V_t

intersection $\{x\} \Cap G \Cap$

the language giving weig

to other sequences in V_s

language giving weight 1

Note that non-null bi-se

their source projection ec

$L_{G,x}$ can be identified wi

over V_t. The intersection

and $L_{G,x}$ is denoted by L

Example Let us cons

SCFG (where again, wei

shown, and where we us

to denote the end of a s

needed for making the gra

the SA automaton of Figu

```
S → NP VP $ / NP VP
NP → ces N A / these
VP → sont A / are A
A → marrons / brown
A → marrons / totall
A → cuits / cooked
A → cuits / finished
N → avocats / avocadoes
N → avocats / lawyers
```

It is easy to see that, for instance, the sentences: *these brown avocadoes are cooked $*, *these brown avocadoes are finished $*, and *these totally corrupt lawyers are finished $* all belong to the intersection $L_{SA,x} \Cap L_{G,x}$, while the sentences *these avocadoes brown are cooked $*, *totally corrupt lawyers are finished these $* belong only to $L_{SA,x}$.

Building the intersection We now describe how to build a W-SCFG that represents the intersection $L_{SA,x} \Cap L_{G,x}$. We base our explanations on the example just given.

A Relaxation of the Intersection At the bottom of Figure 1, we show how we can associate an automaton A_t with the σ-automaton SA: we simply "forget" the source-sides of the labels carried by the transitions, and retain all the weights. As before, note that we are only showing a subset of the transitions here.

All valid paths for SA map into valid paths for A_t (with the same weights), but the reverse is not

Source automaton

This is a *relaxation* of the true intersection, but one that we can represent through a W-SCFG, as we know from section 2.[10]

This being noted, we now move to the construction of the full intersection.

The full intersection We discussed in section 2 how to modify a synchronous grammar rule in order to produce the indexed rule scheme (2) in order to represent the bilateral intersection of the grammar with two automata. Let us redo that construction here, in the case of our example

[10]Note that, in the case of our very simple example, any target string that belongs to this relaxed intersection (which consists of the eight sentences *these* {*brown* | *totally corrupt*} {*avocadoes* | *lawyers*} *are* {*cooked* | *finished*}) actually belongs to the full intersection, as none of these sentences corresponds to a path in SA that violates the token-consumption constraint. More generally, it may often be the case in practice that the W-SCFG, by itself, provides enough "control" of the possible target sentences to prevent generation of sentences that would violate the token-consumption constraints, so that there may be little difference in practice between performing the relaxed intersection $\{x\} \Cap G \Cap A_t$ and performing the full intersection $\{x\} \Cap G \Cap L_{SA,x}$.

W-SCFG, of the target automaton represented on the bottom of Figure 1, and of the source automaton $\{x\}$.

The construction is then done in three steps:

$$
\begin{aligned}
{}_{s_0}\mathrm{NP}_{s_3} &\rightarrow {}_{s_0}\mathrm{ces}_{s_1}\ \ {}_{s_1}\mathrm{N}_{s_2}\ \ {}_{s_2}\mathrm{A}_{s_3}\ \ / \\
&\qquad \text{these}\ \ \mathrm{A}\ \ \mathrm{N} \\
{}^{t_0}\mathrm{NP}^{t_3} &\rightarrow \text{ces}\ \ \mathrm{N}\ \ \mathrm{A}\ \ / \\
&\qquad {}^{t_0}\text{these}^{t_1}\ \ {}^{t_1}\mathrm{A}^{t_2}\ \ {}^{t_2}\mathrm{N}^{t_3} \\
{}^{t_0}_{s_0}\mathrm{NP}^{t_3}_{s_3} &\rightarrow {}_{s_0}\mathrm{ces}_{s_1}\ \ {}^{t_2}_{s_1}\mathrm{N}^{t_3}_{s_2}\ \ {}^{t_1}_{s_2}\mathrm{A}^{t_2}_{s_3}\ \ / \\
&\qquad {}^{t_0}\text{these}^{t_1}\ \ {}^{t_1}_{s_2}\mathrm{A}^{t_2}_{s_3}\ \ {}^{t_2}_{s_1}\mathrm{N}^{t_3}_{s_2}
\end{aligned}
$$

In order to adapt that construction to the case where we want the intersection to be with a σ-automaton, what we need to do is to further specialize the nonterminals. Rather than specializing a nonterminal X in the form ${}^{t}_{s}X^{t'}_{s'}$, we specialize it in the form: ${}^{t}_{s}X^{t',\sigma}_{s'}$, where σ represents a set of source tokens that correspond to "collecting" the source tokens in the σ-automaton along a path connecting the states t and t'.[11]

We then proceed to define a new rule scheme associated to our rule, which is obtained as before in three steps, as follows.

$$
\begin{aligned}
{}_{s_0}\mathrm{NP}_{s_3} &\rightarrow {}_{s_0}\mathrm{ces}_{s_1}\ \ {}_{s_1}\mathrm{N}_{s_2}\ \ {}_{s_2}\mathrm{A}_{s_3}\ \ / \\
&\qquad \text{these}\ \ \mathrm{A}\ \ \mathrm{N} \\
{}^{t_0}\mathrm{NP}^{t_3,\sigma_{03}} &\rightarrow \text{ces}\ \ \mathrm{N}\ \ \mathrm{A}\ \ / \\
&\qquad {}^{t_0}\text{these}^{t_1,\sigma_{01}}\ \ {}^{t_1}\mathrm{A}^{t_2,\sigma_{12}}\ \ {}^{t_2}\mathrm{N}^{t_3,\sigma_{23}} \\
{}^{t_0}_{s_0}\mathrm{NP}^{t_3,\sigma_{03}}_{s_3} &\rightarrow {}_{s_0}\mathrm{ces}_{s_1}\ \ {}^{t_2,\sigma_{23}}_{s_1}\mathrm{N}^{t_3}_{s_2}\ \ {}^{t_1,\sigma_{12}}_{s_2}\mathrm{A}^{t_2}_{s_3}\ \ / \\
&\qquad {}^{t_0}\text{these}^{t_1,\sigma_{01}}\ \ {}^{t_1}_{s_2}\mathrm{A}^{t_2,\sigma_{12}}_{s_3}\ \ {}^{t_2}_{s_1}\mathrm{N}^{t_3,\sigma_{23}}_{s_2}
\end{aligned}
$$

The only difference with our previous technique is in the addition of the σ's to the top indices. Let us focus on the second step of the annotation process:

$$
\begin{aligned}
{}^{t_0}\mathrm{NP}^{t_3,\sigma_{03}} &\rightarrow \text{ces}\ \ \mathrm{N}\ \ \mathrm{A}\ \ / \\
&\qquad {}^{t_0}\text{these}^{t_1,\sigma_{01}}\ \ {}^{t_1}\mathrm{A}^{t_2,\sigma_{12}}\ \ {}^{t_2}\mathrm{N}^{t_3,\sigma_{23}}
\end{aligned}
$$

[11] To avoid a possible confusion, it is important to note right away that σ *is not necessarily related* to the tokens appearing between the positions s and s' in the source sentence (that is, between these states in the associated source automaton), but is defined solely in terms of the source tokens along the t, t' path. See example with "persons" and "people" below.

Conceptually, when instanciating this scheme, the t_i's may range over all possible states of the σ-automaton, and the σ_{ij} over all subsets of the source tokens, but under the following constraints: the RHS σ's (here $\sigma_{01}, \sigma_{12}, \sigma_{23}$) must be disjoint and their union must be equal to the σ on the LHS (here σ_{03}). Additionally, a σ associated with a target terminal (as σ_{01} here) must be equal to the token set associated to the transition that this terminal realizes between σ-automaton states (here, this means that σ_{01} must be equal to the token set $\{ces\}$ associated with the transition between t_0, t_1 labelled with 'these'). If we perform all these instantiations, compute their weights according to equation (3), and finally remove the indices associated with terminals in the rules (by adding the weights of the rules only differing by the indices of terminals, as done previously), we obtain a very large "raw" grammar, but one for which one can prove direct counterpart of **Fact 1**. Let us call, as before G' the raw W-SCFG obtained, its start symbol being ${}^{t_\#}_{s_\#}S^{t_\$,\sigma_{all}}_{s_\$}$, with σ_{all} the set of all source tokens in x.

Fact 2. *The synchronous language $L_{G'}$ associated with G' is equal to* $(\{x\}, L_{SA,x} \Cap L_{G,x})$.

The grammar that is obtained this way, despite correctly representing the intersection, contains a lot of useless rules, this being due to the fact that many nonterminals can not produce any output. The situation is wholly similar to the case of section 2, and the same bottom-up techniques can be used for activating nonterminals and rules bottom-up.

The algorithm is illustrated in Figure 2, where we have shown the result of the process of activating in turn the nonterminals (abbreviated by) N1, A1, A2, NP1, VP1, S1. As a consequence of these activations, the original grammar rule $NP \rightarrow ces\ N\ A\ /these\ A\ N$ (for instance) becomes instantiated as the rule:

$$
\begin{aligned}
&{}^{\#}_{0}\mathrm{NP}^{tcl,\{ces,avocats,marrons\}}_{3} \rightarrow \\
&{}_{0}\mathrm{ces}_{1}\ \ {}^{tcl2}_{1}\mathrm{N}^{tcl,\emptyset}_{2}\ \ {}^{h}_{2}\mathrm{A}^{tcl2,\{avocats,marrons\}}_{3}\ \ / \\
&{}^{\#}\text{these}^{h,\{ces\}}\ \ {}^{h}_{2}\mathrm{A}^{tcl2,\{avocats,marrons\}}_{3}\ \ {}^{tcl2}_{1}\mathrm{N}^{tcl,\emptyset}_{2}
\end{aligned}
$$

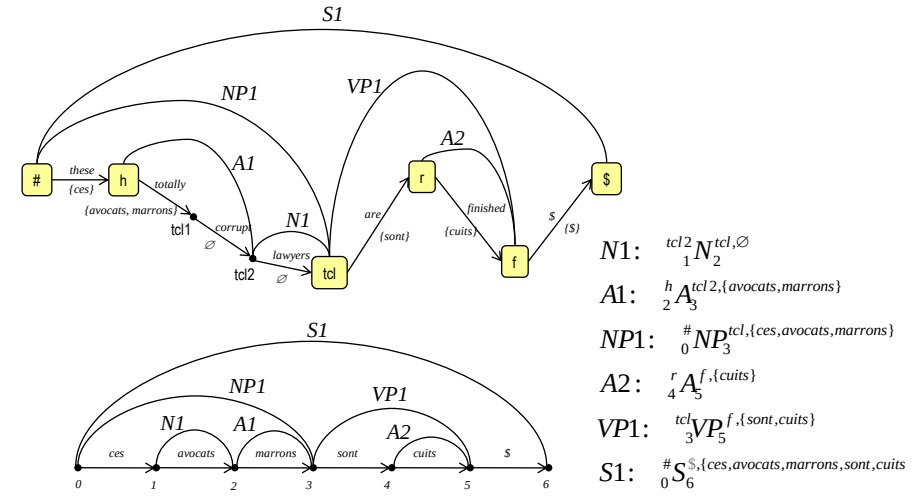


Figure 2: Building the intersection. The bottom of the figure shows some active non-terminals associated with the source sequence, at the top these same non-terminals associated with a sequence of transitions in the σ-automaton, corresponding to the target sequence *these totally corrupt lawyers are finished $.* To avoid cluttering the drawing, we have used the abbreviations shown on the right. Note that while A1 only spans *marrons* in the bottom chart, it is actually decorated with the source token set $\{avocats, marrons\}$: such a "disconnect" between the views that the W-SCFG and the σ-automaton have of the source tokens is not ruled out.

that is, after removal of the indices on terminals:

$$
\begin{aligned}
&{}^{\#}_{0}\mathrm{NP}^{tcl,\{ces,avocats,marrons\}}_{3} \rightarrow \\
&\text{ces} \quad {}^{tcl2}_{1}\mathrm{N}^{tcl,\emptyset}_{2} \quad {}^{h}_{2}\mathrm{A}^{tcl2,\{avocats,marrons\}}_{3} \qquad / \\
&\text{these} \quad {}^{h}_{2}\mathrm{A}^{tcl2,\{avocats,marrons\}}_{3} \quad {}^{tcl2}_{1}\mathrm{N}^{tcl,\emptyset}_{2}
\end{aligned}
$$

Note that while the nonterminal ${}^{tcl2}_{1}\mathrm{N}^{tcl,\emptyset}_{2}$ by itself consumes no source token (it is associated with the empty token set), any actual use of this nonterminal (in this specific rule or possibly in some other rule using it) does require traversing the internal node $tcl2$ and therefore all the internal nodes "belonging" to the biphrase tcl (because otherwise the path from # to $ would be disconnected); in particular this involves consuming all the tokens on the source side of tcl, including 'avocats'.[12]

Complexity considerations The bilateral intersection that we defined between a W-SCFG and two W-FSA's in section 2 can be shown to be of polynomial complexity in the sense that it takes polynomial time and space relative to the sum of the sizes of the two automata and of the grammar to construct the (reduced) intersected grammar G', *under the condition that the grammar right-hand sides have length bounded by a constant.*[13]

The situation here is different, because the construction of the intersection can in principle introduce nonterminals indexed not only by states of the automata, but also by arbitrary subsets of source tokens, and this may lead in extreme cases to an exponential number of rules. Such problems however can only happen in situations where, in a nonterminal ${}^{t}_{s}X^{t',\sigma}_{s'}$, the set σ is allowed to contain tokens that are "unrelated" to the token set $\{personnes\}$ appearing between s and s' in the source automaton. An illustration of such a situation is given by the following example. Suppose that the source sen-

[12] In particular there is no risk that a derivation relative to the intersected grammar generates a target containing two instances of 'lawyers', one associated to the expansion of ${}^{tcl2}_{1}\mathrm{N}^{tcl,\emptyset}_{2}$ and consuming no source token, and another one associated with a different nonterminal and consuming the source token 'avocats': this second instance would involve not traversing $tcl1$, which is impossible as soon as ${}^{tcl2}_{1}\mathrm{N}^{tcl,\emptyset}_{2}$ is used.

[13] If this condition is removed, and for the simpler case where the source (resp. target) automaton encodes a single sentence x (resp. y), (Satta and Peserico, 2005) have shown that the problem of deciding whether (x, y) is recognized by G is NP-hard relative to the sum of the sizes. A consequence is then that the grammar G' cannot be constructed in polynomial time unless $P = NP$.

tence contains the two tokens $personnes$ and $gens$ between positions $i, i+1$ and $j, j+1$ respectively, with i and j far from each other, that the phrase table contains the two phrase pairs $(personnes, persons)$ and $(gens, people)$, but that the synchronous grammar only contains the two rules $X \rightarrow personnes/people$ and $Y \rightarrow gens/persons$, with these phrases and rules exhausting the possibilities for translating $gens$ and $personnes$; Then the intersected grammar will contain such nonterminals as ${}^{t}_{i}X^{t',\{gens\}}_{i+1}$ and ${}^{r}_{j}Y^{r',\{personnes\}}_{j+1}$, where in the first case the token set $\{gens\}$ in the first nonterminal is unrelated to the tokens appearing between $i, i+1$, and similarly in the second case.

Without experimentation on real cases, it is impossible to say whether such phenomena would empirically lead to combinatorial explosion or whether the synchronous grammar would sufficiently constrain the phrase-base component (whose re-ordering capabilities are responsible *in fine* for the potential NP-hardness of the translation process) to avoid it. Another possible approach is to prevent *a priori* a possible combinatorial explosion by adding formal constraints to the intersection mechanism. One such constraint is the following: disallow introduction of ${}^{t}_{i}X^{t',\sigma}_{j}$ when the symmetric difference between σ and the set of tokens between positions i and j in the source sentence has cardinality larger than a small constant. Such a constraint could be interpreted as keeping the SCFG and phrase-base components "in sync", and would be better adapted to the spirit of our approach than limiting the amount of re-ordering permitted to the phrase-based component, which would contradict the reason for using a hierarchical component in the first place.

5 Conclusion

Intersecting hierarchical and phrase-based models of translation could allow to capitalize on complementarities between the two approaches. Thus, one might train the hierarchical component on corpora represented at the part-of-speech level (or at a level where lexical units are abstracted into some kind of classes) while the phrase-based component would focus on translation of lexical material. The present paper does not have the ambition to demonstrate that such an approach would improve translation performance, but only to provide some formal means for advancing towards that goal.

References

Bar-Hillel, Y., M. Perles, and E. Shamir. 1961. On formal properties of simple phrase structure grammars. *Zeitschrift für Phonetik, Sprachwissenschaft und Kommunicationsforschung*, 14:143–172.

Berstel, Jean. 1979. *Transductions and Context-Free Languages*. Teubner, Stuttgart.

Blunsom, P. and M. Osborne. 2008. Probabilistic inference for machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 215–223. Association for Computational Linguistics. Slides downloaded.

Chiang, David. 2006. An introduction to synchronous grammars. `www.isi.edu/~chiang/papers/synchtut.pdf`, June.

Chiang, David. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33:201–228.

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 *ACL*. The Association for Computer Linguistics.

Kumar, Shankar and William Byrne. 2005. Local phrase reordering models for statistical machine translation. In *Proc. HLT/EMNLP*.

Lang, Bernard. 1994. Recognition can be harder than parsing. *Computational Intelligence*, 10:486–494.

Mohri, Mehryar, Fernando Pereira, and Michael Riley. 1996. Weighted automata in text and speech processing. In *ECAI-96 Workshop on Extended Finite State Models of Language*.

Mohri, Mehryar. 2002. Generic epsilon-removal and input epsilon-normalization algorithms for weighted transducers. *International Journal of Foundations of Computer Science*, 13:129–143.

Och, Franz Josef and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Comput. Linguist.*, 30(4):417–449.

Satta, Giorgio and Enoch Peserico. 2005. Some computational complexity results for synchronous context-free grammars. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 803–810, Morristown, NJ, USA. Association for Computational Linguistics.

Satta, Giorgio. 2008. Translation algorithms by means of language intersection. Submitted. `www.dei.unipd.it/~satta/publ/paper/inters.pdf`.

A Systematic Comparison between Inversion Transduction Grammar and Linear Transduction Grammar for Word Alignment

Markus Saers and **Joakim Nivre**
Dept. of Linguistics & Philology
Uppsala University
first.last@lingfil.uu.se

Dekai Wu
HKUST
Human Language Technology Center
Dept. of Computer Science & Engineering
Hong Kong Univ. of Science & Technology
dekai@cs.ust.hk

Abstract

We present two contributions to grammar driven translation. First, since both Inversion Transduction Grammar and Linear Inversion Transduction Grammars have been shown to produce better alignments then the standard word alignment tool, we investigate how the trade-off between speed and end-to-end translation quality extends to the choice of grammar formalism. Second, we prove that Linear Transduction Grammars (LTGs) generate the same transductions as Linear Inversion Transduction Grammars, and present a scheme for arriving at LTGs by bilingualizing Linear Grammars. We also present a method for obtaining Inversion Transduction Grammars from Linear (Inversion) Transduction Grammars, which can speed up grammar induction from parallel corpora dramatically.

1 Introduction

In this paper we introduce Linear Transduction Grammars (LTGs), which are the bilingual case of Linear Grammars (LGs). We also show that LTGs are equal to Linear Inversion Transduction Grammars (Saers et al., 2010). To be able to induce transduction grammars directly from parallel corpora an approximate search for parses is needed. The trade-off between speed and end-to-end translation quality is investigated and compared to Inversion Transduction Grammars (Wu, 1997) and the standard tool for word alignment, GIZA++ (Brown et al., 1993; Vogel et al., 1996; Och and Ney, 2003). A heuristic for converting stochastic bracketing LTGs into stochastic bracketing ITGs is presented, and fitted into the speed–quality trade-off.

In section 3 we give an overview of transduction grammars, introduce LTGs and show that they are equal to LITGs. In section 4 we give a short description of the rational for the transduction grammar pruning used. In section 5 we describe a way of seeding a stochastic bracketing ITG with the rules and probabilities of a stochastic bracketing LTG. Section 6 describes the setup, and results are given in section 7. Finally, some conclusions are offered in section 8

2 Background

Any form of automatic translation that relies on generalizations of observed translations needs to align these translations on a sub-sentential level. The standard way of doing this is by aligning words, which works well for languages that use white space separators between words. The standard method is a combination of the family of IBM-models (Brown et al., 1993) and Hidden Markov Models (Vogel et al., 1996). These methods all arrive at a function (A) from language 1 (F) to language 2 (E). By running the process in both directions, two functions can be estimated and then combined to form an alignment. The simplest of these combinations are intersection and union, but usually, the intersection is heuristically extended. Transduction grammars on the other hand, impose a shared structure on the sentence pairs, thus forcing a consistent alignment in both directions. This method

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 10–18,
COLING 2010, Beijing, August 2010.

has proved successful in the settings it has been tried (Zhang et al., 2008; Saers and Wu, 2009; Haghighi et al., 2009; Saers et al., 2009; Saers et al., 2010). Most efforts focus on cutting down time complexity so that larger data sets than toy-examples can be processed.

3 Transduction Grammars

Transduction grammars were first introduced in Lewis and Stearns (1968), and further developed in Aho and Ullman (1972). The original notation called for regular CFG-rules in language F with rephrased E productions, either in curly brackets, or comma separated. The bilingual version of CFGs is called Syntax-Directed Transduction Grammars (SDTGs). To differentiate identical nonterminal symbols, indices were used (the bag of nonterminals for the two productions are equal by definition).

$$\begin{aligned} &A \rightarrow B^{(1)}\ a\ B^{(2)} \quad \{x\ B^{(1)}\ B^{(2)}\} \\ &= A \rightarrow B^{(1)}\ a\ B^{(2)},\ x\ B^{(1)}\ B^{(2)} \end{aligned}$$

The semantics of the rules is that one nonterminal rewrites into a bag of nonterminals that is distributed independently in the two languages, and interspersed with any number of terminal symbols in the respective languages. As with CFGs, the terminal symbols can be factored out into preterminals with the added twist that they are shared between the two languages, since preterminals are formally nonterminals. The above rule can thus be rephrased as

$$\begin{aligned} A &\rightarrow B^{(1)}\ X^{a/x}\ B^{(2)},\ X^{a/x}\ B^{(1)}\ B^{(2)} \\ X^{a/x} &\rightarrow a,\ x \end{aligned}$$

In this way, rules producing nonterminals and rules producing terminals can be separated. Since only nonterminals are allowed to move, their movement can be represented as the original sequence of nonterminals and a permutation vector as follows:

$$\begin{aligned} A &\rightarrow B\ X^{a/x}\ B\ ;\ 1,0,2 \\ X^{a/x} &\rightarrow a,\ x \end{aligned}$$

To keep the reordering as monotone as possible, the terminals a and x can be produced separately, but doing so eliminates any possibility of parameterizing their lexical relationship. Instead, the individual terminals are pair up with the empty string (ϵ).

$$\begin{aligned} A &\rightarrow X^{x}\ B\ X^{a}\ B\ ;\ 0,1,2,3 \\ X^{a} &\rightarrow a,\ \epsilon \\ X^{x} &\rightarrow \epsilon,\ x \end{aligned}$$

Lexical rules involving the empty string are referred to as singletons. Whenever a preterminal is used to pair up two terminal symbols, we refer to that pair of terminals as a *biterminal*, which will be written as e/f.

Any SDTG can be rephrased to contain permuted nonterminal productions and biterminal productions only, and we will call this the normal form of SDTGs. Note that it is not possible to produce a two-normal form for SDTGs, as there are some rules that are not binarizable (Wu, 1997; Huang et al., 2009). This is an important point to make, since efficient parsing for CFGs is based on either restricting parsing to only handle binary grammars (Cocke, 1969; Kasami, 1965; Younger, 1967), or rely on on-the-fly binarization (Earley, 1970). When translating with a grammar, parsing only has to be done in F, which is binarizable (since it is a CFG), and can therefor be computed in polynomial time ($\mathcal{O}(n^3)$). Once there is a parse tree for F, the corresponding tree for E can be easily constructed. When inducing a grammar from examples, however, biparsing (finding an analysis that is consistent across a sentence pair) is needed. The time complexity for biparsing with SDTGs is $\mathcal{O}(n^{2n+2})$, which is clearly intractable.

Inversion Transduction Grammars or ITGs (Wu, 1997) are transduction grammars that have a two-normal form, thus guaranteeing binarizability. Defining the rank of a rule as the number of nonterminals in the production, and the rank of a grammar as the highest ranking rule in the rule set, ITGs are *a)* any SDTG of rank two, *b)* any SDTG of rank three or *c)* any SDTG where no rule has a permutation vector other than identity permutation or inversion permutation. It follows from this definition that ITGs have a two-normal form, which is usually expressed as SDTG rules,

with brackets around the production to distinguish the different kinds of rules from each other.

$$\begin{array}{lcl} A \rightarrow B\ C\ ;\ 0,1 & = & A \rightarrow [\ B\ C\ ] \\ A \rightarrow B\ C\ ;\ 1,0 & = & A \rightarrow \langle\ B\ C\ \rangle \\ A \rightarrow e/f & = & A \rightarrow e/f \end{array}$$

By guaranteeing binarizability, biparsing time complexity becomes $\mathcal{O}(n^6)$.

There is an even more restricted version of SDTGs called Simple Transduction Grammar (STG), where no permutation at all is allowed, which can also biparse a sentence pair in $\mathcal{O}(n^6)$ time.

A Linear Transduction Grammar (LTG) is a bilingual version of a Linear Grammar (LG).

Definition 1. An LG in normal form is a tuple

$$\mathcal{G}_L = \langle N, \Sigma, R, S \rangle$$

Where N is a finite set of nonterminal symbols, Σ is a finite set of terminal symbols, R is a finite set of rules and $S \in N$ is the designated start symbol. The rule set is constrained so that

$$R \subseteq N \times (\Sigma \cup \{\epsilon\}) N (\Sigma \cup \{\epsilon\}) \cup \{\epsilon\}$$

Where ϵ is the empty string.

To bilingualize a linear grammar, we will take the same approach as taken when a finite-state automaton is bilingualized into a finite-state transducer. That is: to replace all terminal symbols with biterminal symbols.

Definition 2. An LTG in normal form is a tuple

$$\mathcal{TG}_L = \langle N, \Sigma, \Delta, R, S \rangle$$

Where N is a finite set of nonterminal symbols, Σ is a finite set of terminal symbols in language E, Δ is a finite set of terminal symbols in language F, R is a finite set of linear transduction rules and $S \in N$ is the designated start symbol. The rule set is constrained so that

$$R \subseteq N \times \Psi N \Psi \cup \{\langle \epsilon, \epsilon \rangle\}$$

Where $\Psi = \Sigma \cup \{\epsilon\} \times \Delta \cup \{\epsilon\}$ and ϵ is the empty string.

Graphically, we will represent LTG rules as production rules with biterminals:

$$\begin{array}{rcl} \langle A, \langle x, p \rangle B \langle y, q \rangle \rangle & = & A \rightarrow x/p\ B\ y/q \\ \langle A, \langle \epsilon, \epsilon \rangle \rangle & = & B \rightarrow \epsilon/\epsilon \end{array}$$

Like STGs, LTGs do not allow any reordering, and are monotone, but because they are linear, this has no impact on expressiveness, as we shall see later.

Linear Inversion Transduction Grammars (LITGs) were introduced in Saers et al. (2010), and represent ITGs that are allowed to have at most one nonterminal symbol in each production. These are attractive because they can biparse a sentence pair in $\mathcal{O}(n^4)$ time, which can be further reduced to linear time by severely pruning the search space. This makes them tractable for large parallel corpora, and a viable way to induce transduction grammars from large parallel corpora.

Definition 3. An LITG in normal form is a tuple

$$\mathcal{TG}_{LI} = \langle N, \Sigma, \Delta, R, S \rangle$$

Where N is a finite set of nonterminal symbols, Σ is a finite set of terminal symbols from language E, Δ is a finite set of terminal symbols from language F, R is a set of rules and $S \in N$ is the designated start symbol. The rule set is constrained so that

$$R \subseteq N \times \{[], \langle\rangle\} \times \Psi N \cup N \Psi \cup \{\langle \epsilon, \epsilon \rangle\}$$

Where [] represents identity permutation and $\langle\rangle$ represents inversion permutation, $\Psi = \Sigma \cup \{\epsilon\} \times \Delta \cup \{\epsilon\}$ is a possibly empty biterminal, and ϵ is the empty string.

Graphically, a rule will be represented as an ITG rule:

$$\begin{array}{rcl} \langle A, [], B \langle e, f \rangle \rangle & = & A \rightarrow [\ B\ e/f\ ] \\ \langle A, \langle\rangle, \langle e, f \rangle B \rangle & = & A \rightarrow \langle\ e/f\ B\ \rangle \\ \langle A, [], \langle \epsilon, \epsilon \rangle \rangle & = & A \rightarrow \epsilon/\epsilon \end{array}$$

As with ITGs, productions with only biterminals will be represented without their permutation, as any such rule can be trivially rewritten into inverted or identity form.

Definition 4. An ϵ-free LITG is an LITG where no rule may rewrite one nonterminal into another nonterminal only. Formally, the rule set is constrained so that

$$R \cap N \times \{[], \langle\rangle\} \times (\{\langle\epsilon,\epsilon\rangle\}B \cup B\{\langle\epsilon,\epsilon\rangle\}) = \emptyset$$

The LITG presented in Saers et al. (2010) is thus an ϵ-free LITG in normal form, since it has the following thirteen rule forms (of which 8 are meaningful, 1 is only used to terminate generation and 4 are redundant):

$$\begin{array}{lcl}
A \to [\ e/f\ B\ ] & & \\
A \to \langle\ e/f\ B\ \rangle & & \\
A \to [\ B\ e/f\ ] & & \\
A \to \langle\ B\ e/f\ \rangle & & \\
A \to [\ e/\epsilon\ B\ ] & | & A \to \langle\ e/\epsilon\ B\ \rangle \\
A \to [\ B\ e/\epsilon\ ] & | & A \to \langle\ B\ e/\epsilon\ \rangle \\
A \to [\ \epsilon/f\ B\ ] & | & A \to \langle\ B\ \epsilon/f\ \rangle \\
A \to [\ B\ \epsilon/f\ ] & | & A \to \langle\ \epsilon/f\ B\ \rangle \\
A \to \epsilon/\epsilon & &
\end{array}$$

All the singleton rules can be expressed either in straight or inverted form, but the result of applying the two rules are the same.

Lemma 1. *Any* LITG *in normal form can be expressed as an* LTG *in normal form.*

Proof. The above LITG can be rewritten in LTG form as follows:

$$\begin{array}{lcl}
A \to [\ e/f\ B\ ] & = & A \to e/f\ B \\
A \to \langle\ e/f\ B\ \rangle & = & A \to e/\epsilon\ B\ \epsilon/f \\
A \to [\ B\ e/f\ ] & = & A \to B\ e/f \\
A \to \langle\ B\ e/f\ \rangle & = & A \to \epsilon/f\ B\ e/\epsilon \\
A \to [\ e/\epsilon\ B\ ] & = & A \to e/\epsilon\ B \\
A \to [\ B\ e/\epsilon\ ] & = & A \to B\ e/\epsilon \\
A \to [\ \epsilon/f\ B\ ] & = & A \to \epsilon/f\ B \\
A \to [\ B\ \epsilon/f\ ] & = & A \to B\ \epsilon/f \\
A \to \epsilon/\epsilon & = & A \to \epsilon/\epsilon
\end{array}$$

To account for all LITGs in normal form, the following two non-ϵ-free rules also needs to be accounted for:

$$\begin{array}{lcl}
A \to [\ B\ ] & = & A \to B \\
A \to \langle\ B\ \rangle & = & A \to B
\end{array}$$

□

Lemma 2. *Any* LTG *in normal form can be expressed as an* LITG *in normal form.*

Proof. An LTG in normal form has two rules, which can be rewritten in LITG form, either as straight or inverted rules as follows

$$\begin{array}{lcl}
A \to x/p\ B\ y/q & = & A \to [\ x/p\ \bar{B}\ ] \\
 & & \bar{B} \to [\ B\ y/q\ ] \\
 & = & A \to \langle\ x/q\ \bar{B}\ \rangle \\
 & & \bar{B} \to \langle\ B\ y/p\ \rangle \\
A \to \epsilon/\epsilon & = & A \to \epsilon/\epsilon
\end{array}$$

□

Theorem 1. LTGs *in normal form and* LITGs *in normal form express the same class of transductions.*

Proof. Follows from lemmas 1 and 2. □

By theorem 1 everything concerning LTGs is also applicable to LITGs, and an LTG can be expressed in LITG form when convenient, and vice versa.

4 Pruning the Alignment Space

The alignment space for a transduction grammar is the combinations of the parse spaces of the sentence pair. Let $\mathbf{e}$ be the E sentence, and $\mathbf{f}$ be the F sentence. The parse spaces would be $\mathcal{O}(|\mathbf{e}|^2)$ and $\mathcal{O}(|\mathbf{f}|^2)$ respectively, and the combination of these spaces would be $\mathcal{O}(|\mathbf{e}|^2 \times |\mathbf{f}|^2)$, or $\mathcal{O}(n^4)$ if we assume n to be proportional to the sentence lengths. In the case of LTGs, this space is searched linearly, giving time complexity $\mathcal{O}(n^4)$, and in the case of ITGs there is branching within both parse spaces, adding an order of magnitude each, giving a total time complexity of $\mathcal{O}(n^6)$. There is, in other words, a tight connection between the alignment space and the time complexity of the biparsing algorithm. Furthermore, most of this alignment space is clearly useless. Consider the case where the entire F sentence is deleted, and the entire E sentence is simply inserted. Although it is possible that it is allowed by the grammar, it should have a negligible probability (since it is clearly a translation strategy that generalize poorly), and could, for all practical reasons, be ignored.

Language pair	Bisentences	Tokens
Spanish–English	108,073	1,466,132
French–English	95,990	1,340,718
German–English	115,323	1,602,781

Table 1: Size of training data.

Saers et al. (2009) present a scheme for pruning away most of the points in the alignment space. Parse items are binned according to coverage (the total number of words covered), and each bin is restricted to carry a maximum of b items. Any items that do not fit in the bins are excluded from further analysis. To decide which items to keep, inside probability is used. This pruning scheme effectively linearizes the alignment space, as is will be of size $\mathcal{O}(nb)$, regardless of what type grammar is used. An ITG can thus be biparsed in cubic time, and an LTG in linear time.

5 Seeding an ITG with an LTG

Since LTGs are a subclass of ITGs, it would be possible to convert an LTG to a ITG. This could save a lot of time, since LTGs are *much* faster to induce from corpora than ITGs.

Converting a BLTG to a BITG is fairly straight forward. Consider the BLTG rule

$$X \to [\ e/f\ X\ ]$$

To convert it to BITG in two-normal form, the biterminal has to be factored out. Replacing the biterminal with a temporary symbol $\bar{X}$, and introducing a rule that rewrites this temporary symbol to the replaced biterminal produces two rules:

$$X \to [\ \bar{X}\ X\ ]$$
$$\bar{X} \to e/f$$

This is no longer a bracketing grammar since there are two nonterminals, but equating $\bar{X}$ to X restores this property. An analogous procedure can be applied in the case where the nonterminal comes before the biterminal, as well as for the inverting cases.

When converting stochastic LTGs, the probability mass of the SLTG rule has to be distributed to two SITG rules. The fact that the LTG rule $X \to \epsilon/\epsilon$ lacks correspondence in ITGs has to be weighted in as well. In this paper we took the maximum entropy approach and distributed the probability mass uniformly. This means defining the probability mass function p' for the new SBITG from the probability mass function p of the original SBLTG such that:

$$p'(X \to [\ X\ X\ ]) = \sum_{e/f} \left[\begin{array}{c} \sqrt{\frac{p(X \to [\ e/f\ X\ ])}{1-p(X \to \epsilon/\epsilon)}} \\ + \\ \sqrt{\frac{p(X \to [\ X\ e/f\ ])}{1-p(X \to \epsilon/\epsilon)}} \end{array} \right]$$

$$p'(X \to \langle\ X\ X\ \rangle) = \sum_{e/f} \left[\begin{array}{c} \sqrt{\frac{p(X \to \langle\ e/f\ X\ \rangle)}{1-p(X \to \epsilon/\epsilon)}} \\ + \\ \sqrt{\frac{p(X \to \langle\ X\ e/f\ \rangle)}{1-p(X \to \epsilon/\epsilon)}} \end{array} \right]$$

$$p'(X \to e/f) = \left[\begin{array}{c} \sqrt{\frac{p(X \to [\ e/f\ X\ ])}{1-p(X \to \epsilon/\epsilon)}} \\ + \\ \sqrt{\frac{p(X \to [\ X\ e/f\ ])}{1-p(X \to \epsilon/\epsilon)}} \\ + \\ \sqrt{\frac{p(X \to \langle\ e/f\ X\ \rangle)}{1-p(X \to \epsilon/\epsilon)}} \\ + \\ \sqrt{\frac{p(X \to \langle\ X\ e/f\ \rangle)}{1-p(X \to \epsilon/\epsilon)}} \end{array} \right]$$

6 Setup

The aim of this paper is to compare the alignments from SBITG and SBLTG to those from GIZA++, and to study the impact of pruning on efficiency and translation quality. Initial grammars will be estimated by counting cooccurrences in the training corpus, after which expectation-maximization (EM) will be used to refine the initial estimate. At the last iteration, the one-best parse of each sentence will be considered as the word alignment of that sentence.

In order to keep the experiments comparable, relatively small corpora will be used. If larger corpora were used, it would not be possible to get any results for unpruned SBITGs because of the prohibitive time complexity. The Europarl corpus (Koehn, 2005) was used as a starting point, and then all sentence pairs where one of the sentences were longer than 10 tokens were filtered

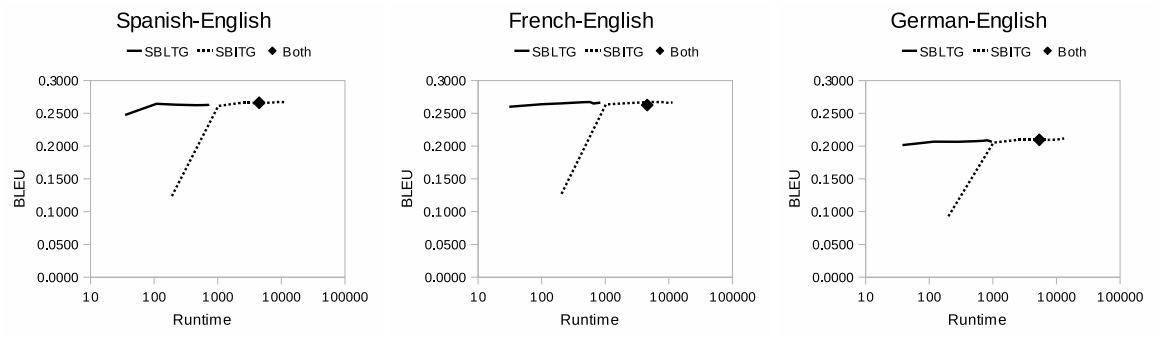


Figure 1: Trade-offs between translation quality (as measured by BLEU) and biparsing time (in seconds plotted on a logarithmic scale) for SBLTGs, SBITGs and the combination.

System	Beam size 1	10	25	50	75	100	∞
BLEU							
SBITG	0.1234	0.2608	**0.2655**	**0.2653**	**0.2661**	**0.2671**	**0.2663**
SBLTG	0.2574	**0.2645**	0.2631	0.2624	0.2625	0.2633	0.2628
GIZA++	**0.2597**	0.2597	0.2597	0.2597	0.2597	0.2597	0.2597
NIST							
SBITG	3.9705	6.6439	**6.7312**	**6.7101**	**6.7329**	**6.7445**	**6.6793**
SBLTG	6.6023	**6.6800**	6.6657	6.6637	6.6714	6.6863	6.6765
GIZA++	**6.6464**	6.6464	6.6464	6.6464	6.6464	6.6464	6.6464
Training times							
SBITG	03:10	17:00	38:00	1:20:00	2:00:00	2:40:00	3:20:00
SBLTG	35	1:49	3:40	7:33	9:44	12:13	11:59

Table 2: Results for the Spanish–English translation task.

out (see table 1). The GIZA++ system was built according to the instructions for creating a baseline system for the Fifth Workshop on Statistical Machine Translation (WMT'10),[1] but the above corpora were used instead of those supplied by the workshop. This includes word alignment with GIZA++, a 5-gram language model built with SRILM (Stolcke, 2002) and parameter tuning with MERT (Och, 2003). To carry out the actual translations, Moses (Koehn et al., 2007) was used. The SBITG and SBLTG systems were built in exactly the same way, except that the alignments from GIZA++ were replaced by those from the respective grammars.

In addition to trying out exhaustive biparsing for SBITGs and SBLTGs on three different translation tasks, several different levels of pruning were tried (1, 10, 25, 50, 75 and 100). We also used the grammar induced from SBLTGs with a beam size of 25 to seed SBITGs (see section 5), which were then run for an additional iteration of EM, also with beam size 25.

All systems are evaluated with BLEU (Papineni et al., 2002) and NIST (Doddington, 2002).

7 Results

The results for the three different translation tasks are presented in Tables 2, 3 and 4. It is interesting to note that the trend they portray is quite similar. When the beam is very narrow, GIZA++ is better, but already at beam size 10, both transduction grammars are superior. Con-

[1] http://www.statmt.org/wmt10/

System	Beam size						
	1	10	25	50	75	100	∞
BLEU							
SBITG	0.1268	0.2632	**0.2654**	**0.2669**	0.2668	0.2655	**0.2663**
SBLTG	0.2600	**0.2638**	0.2651	0.2668	**0.2672**	**0.2662**	0.2649
GIZA++	**0.2603**	0.2603	0.2603	0.2603	0.2603	0.2603	0.2603
NIST							
SBITG	4.0849	6.7136	**6.7913**	**6.8065**	**6.8068**	**6.8088**	**6.8151**
SBLTG	6.6814	**6.7608**	6.7656	6.7992	6.8020	6.7925	6.7784
GIZA++	**6.6907**	6.6907	6.6907	6.6907	6.6907	6.6907	6.6907
Training times							
SBITG	03:25	17:00	42:00	1:25:00	2:10:00	2:45:00	3:10:00
SBLTG	31	1:41	3:25	7:06	9:35	13:56	10:52

Table 3: Results for the French–English translation task.

System	Beam size						
	1	10	25	50	75	100	∞
BLEU							
SBITG	0.0926	0.2050	**0.2091**	**0.2090**	**0.2091**	**0.2094**	**0.2113**
SBLTG	0.2015	**0.2067**	0.2066	0.2073	0.2080	0.2066	0.2088
GIZA++	**0.2059**	0.2059	0.2059	0.2059	0.2059	0.2059	0.2059
NIST							
SBITG	3.4297	5.8743	**5.9292**	5.8947	5.8955	**5.9086**	**5.9380**
SBLTG	5.7799	**5.8819**	5.8882	**5.8963**	**5.9252**	5.8757	5.9311
GIZA++	**5.8668**	5.8668	5.8668	5.8668	5.8668	5.8668	5.8668
Training times							
SBITG	03:20	17:00	41:00	1:25:00	2:10:00	2:45:00	3:40:00
SBLTG	38	1:58	4:52	8:08	11:42	16:05	13:32

Table 4: Results for the German–English translation task.

sistent with Saers et al. (2009), SBITG has a sharp rise in quality going from beam size 1 to 10, and then a gentle slope up to beam size 25, after which it levels out. SBLTG, on the other hand start out at a respectable level, and goes up a gentle slope from beam size 1 to 10, after which is level out. This is an interesting observation, as it suggests that SBLTG reaches its optimum with a lower beam size (although that optimum is lower than that of SBITG). The trade-off between quality and time can now be extended beyond beam size to include grammar choice. In Figure 1, run times are plotted against BLEU scores to illustrate this trade-off. It is clear that SBLTGs are indeed much faster than SBITGs, the only exception is when SBITGs are run with $b = 1$, but then the BLEU score is so low that is is not worth considering.

The time may seem inconsistent between $b = 100$ and $b = \infty$ for SBLTG, but the extra time for the tighter beam is because of beam management, which the exhaustive search doesn't bother with.

In table 5 we compare the pure approaches to one where an LTG was trained during 10 iterations of EM and then used to seed (see sec-

Translation task	System	BLEU	NIST	Total time
Spanish–English	SBLTG	0.2631	6.6657	36:40
	SBITG	0.2655	**6.7312**	6:20:00
	Both	**0.2660**	6.7124	1:14:40
French–English	SBLTG	0.2651	6.7656	34:10
	SBITG	**0.2654**	**6.7913**	7:00:00
	Both	0.2625	6.7609	1:16:10
German–English	SBLTG	0.2066	5.8882	48:52
	SBITG	0.2091	**5.9292**	6:50:00
	Both	**0.2095**	5.9224	1:29:40

Table 5: Results for seeding an SBITG with an SBLTG (Both) compared to the pure approach. Total time refers to 10 iterations of EM training for SBITG and SBLTG respectively, and 10 iterations of SBLTG and one iteration of SBITG training for the combined system.

tion 5) an SBITG, which was then trained for one iteration of EM. Although the differences are fairly small, German–English and Spanish–English seem to reach the level of SBITG, whereas French–English is actually hurt. The big difference is in time, since the combined system needs about a fifth of the time the SBITG-based system needs. This phenomenon needs to be more thoroughly examined.

It is also worth noting that GIZA++ was beaten by an aligner that used less than 20 minutes (less than 2 minutes per iteration and at most 10 iterations) to align the corpus.

8 Conclusions

In this paper we have introduced the bilingual version of linear grammar: Linear Transduction Grammars, and found that they generate the same class of transductions as Linear Inversion Transduction Grammars. We have also compared Stochastic Bracketing versions of ITGs and LTGs to GIZA++ on three word alignment tasks. The efficiency issues with transduction grammars have been addressed by pruning, and the conclusion is that there is a trade-off between run time and translation quality. A part of the trade-off is choosing which grammar framework to use, as LTGs are faster but not as good as ITGs. It also seems possible to take a short-cut in this trade-off by starting out with an LTG and converting it to an ITG. We have also showed that it is possible to beat the translation quality of GIZA++ with a quite fast transduction grammar.

Acknowledgments

This work was funded by the Swedish National Graduate School of Language Technology (GSLT), the Defense Advanced Research Projects Agency (DARPA) under GALE Contracts No. HR0011-06-C-0022 and No. HR0011-06-C-0023, and the Hong Kong Research Grants Council (RGC) under research grants GRF621008, DAG03/04.EG09, RGC6256/00E, and RGC6083/99E. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views ofthe Defense Advanced Research Projects Agency. The computations were performed on UPPMAX resources under project p2007020.

References

Aho, Alfred V. Ullman, Jeffrey D. 1972. *The Theory of Parsing, Translation, and Compiling*. Prentice-Halll, Inc., Upper Saddle River, NJ.

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.

Cocke, John. 1969. *Programming languages and their compilers: Preliminary notes*. Courant Insti-

tute of Mathematical Sciences, New York University.

Doddington, George. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of Human Language Technology conference (HLT-2002)*, San Diego, California.

Earley, Jay. 1970. An efficient context-free parsing algorithm. *Communications of the Association for Comuter Machinery*, 13(2):94–102.

Haghighi, Aria, John Blitzer, John DeNero, and Dan Klein. 2009. Better word alignments with supervised ITG models. 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 923–931, Suntec, Singapore, August.

Huang, Liang, Hao Zhang, Daniel Gildea, and Kevin Knight. 2009. Binarization of synchronous context-free grammars. *Computational Linguistics*, 35(4):559–595.

Kasami, Tadao. 1965. An efficient recognition and syntax analysis algorithm for context-free languages. Technical Report AFCRL-65-00143, Air Force Cambridge Research Laboratory.

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 *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *Machine Translation Summit X*, Phuket, Thailand, September.

Lewis, Philip M. and Richard E. Stearns. 1968. Syntax-directed transduction. *Journal of the Association for Computing Machinery*, 15(3):465–488.

Och, Franz Josef and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Och, Franz Josef. 2003. Minimum error rate training in statistical machine translation. In *41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan, July.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, July.

Saers, Markus and Dekai Wu. 2009. Improving phrase-based translation via word alignments from Stochastic Inversion Transduction Grammars. In *Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation (SSST-3) at NAACL HLT 2009*, pages 28–36, Boulder, Colorado, June.

Saers, Markus, Joakim Nivre, and Dekai Wu. 2009. Learning Stochastic Bracketing Inversion Transduction Grammars with a cubic time biparsing algorithm. In *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*, pages 29–32, Paris, France, October.

Saers, Markus, Joakim Nivre, and Dekai Wu. 2010. Word alignment with Stochastic Bracketing Linear Inversion Transduction Grammar. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, California, June.

Stolcke, Andreas. 2002. SRILM – an extensible language modeling toolkit. In *International Conference on Spoken Language Processing*, Denver, Colorado, September.

Vogel, Stephan, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics*, pages 836–841, Morristown, New Jersey.

Wu, Dekai. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Younger, Daniel H. 1967. Recognition and parsing of context-free languages in time n^3. *Information and Control*, 10(2):189–208.

Zhang, Hao, Chris Quirk, Robert C. Moore, and Daniel Gildea. 2008. Bayesian learning of non-compositional phrases with synchronous parsing. In *Proceedings of ACL-08: HLT*, pages 97–105, Columbus, Ohio, June.

Source-side Syntactic Reordering Patterns with Functional Words for Improved Phrase-based SMT

Jie Jiang, Jinhua Du, Andy Way
CNGL, School of Computing, Dublin City University
{jjiang,jdu,away}@computing.dcu.ie

Abstract

Inspired by previous source-side syntactic reordering methods for SMT, this paper focuses on using automatically learned syntactic reordering patterns with functional words which indicate structural reorderings between the source and target language. This approach takes advantage of phrase alignments and source-side parse trees for pattern extraction, and then filters out those patterns without functional words. Word lattices transformed by the generated patterns are fed into PB-SMT systems to incorporate potential reorderings from the inputs. Experiments are carried out on a medium-sized corpus for a Chinese–English SMT task. The proposed method outperforms the baseline system by 1.38% relative on a randomly selected testset and 10.45% relative on the NIST 2008 testset in terms of BLEU score. Furthermore, a system with just 61.88% of the patterns filtered by functional words obtains a comparable performance with the unfiltered one on the randomly selected testset, and achieves 1.74% relative improvements on the NIST 2008 testset.

1 Introduction

Previous work has shown that the problem of structural differences between language pairs in SMT can be alleviated by source-side syntactic reordering. Taking account for the integration with SMT systems, these methods can be divided into two different kinds of approaches (Elming, 2008): the deterministic reordering and the non-deterministic reordering approach.

To carry out the deterministic approach, syntactic reordering is performed uniformly on the training, devset and testset before being fed into the SMT systems, so that only the reordered source sentences are dealt with while building during the SMT system. In this case, most work is focused on methods to extract and to apply syntactic reordering patterns which come from manually created rules (Collins et al., 2005; Wang et al., 2007a), or via an automatic extraction process taking advantage of parse trees (Collins et al., 2005; Habash, 2007). Because reordered source sentence cannot be undone by the SMT decoders (Al-Onaizan et al., 2006), which implies a systematic error for this approach, classifiers (Chang et al., 2009b; Du & Way, 2010) are utilized to obtain high-performance reordering for some specialized syntactic structures (e.g. *DE* construction in Chinese).

On the other hand, the non-deterministic approach leaves the decisions to the decoders to choose appropriate source-side reorderings. This is more flexible because both the original and reordered source sentences are presented in the inputs. Word lattices generated from syntactic structures for N-gram-based SMT is presented in (Crego et al., 2007). In (Zhang et al., 2007a; Zhang et al., 2007b), chunks and POS tags are used to extract reordering rules, while the generated word lattices are weighted by language models and reordering models. Rules created from a syntactic parser are also utilized to form weighted n-best lists which are fed into the decoder (Li et al., 2007). Furthermore, (Elming, 2008; Elm-

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 19–27,
COLING 2010, Beijing, August 2010.

ing, 2009) uses syntactic rules to score the output word order, both on English–Danish and English–Arabic tasks. Syntactic reordering information is also considered as an extra feature to improve PBSMT in (Chang et al., 2009b) for the Chinese–English task. These results confirmed the effectiveness of syntactic reorderings.

However, for the particular case of Chinese source inputs, although the *DE* construction has been addressed for both PBSMT and HPBSMT systems in (Chang et al., 2009b; Du & Way, 2010), as indicated by (Wang et al., 2007a), there are still lots of unexamined structures that imply source-side reordering, especially in the non-deterministic approach. As specified in (Xue, 2005), these include the *bei*-construction, *ba*-construction, three kinds of *de*-construction (including *DE* construction) and general preposition constructions. Such structures are referred with functional words in this paper, and all the constructions can be identified by their corresponding tags in the Penn Chinese TreeBank. It is interesting to investigate these functional words for the syntactic reordering task since most of them tend to produce structural reordering between the source and target sentences.

Another related work is to filter the bilingual phrase pairs with closed-class words (Sánchez-Martínez, 2009). By taking account of the word alignments and word types, the filtering process reduces the phrase tables by up to a third, but still provide a system with competitive performance compared to the baseline. Similarly, our idea is to use special type of words for the filtering purpose on the syntactic reordering patterns.

In this paper, our objective is to exploit these functional words for source-side syntactic reordering of Chinese–English SMT in the non-deterministic approach. Our assumption is that syntactic reordering patterns with functional words are the most effective ones, and others can be pruned for both speed and performance.

To validate this assumption, three systems are compared in this paper: a baseline PBSMT system, a syntactic reordering system with all patterns extracted from a corpus, and a syntactic reordering system with patterns filtered with functional words. To accomplish this, firstly the lattice scoring approach (Jiang et al., 2010) is utilized to discover non-monotonic phrase alignments, and then syntactic reordering patterns are extracted from source-side parse trees. After that, functional word tags specified in (Xue, 2005) are adopted to perform pattern filtering. Finally, both the unfiltered pattern set and the filtered one are used to transform inputs into word lattices to present potential reorderings for improving PBSMT system. A comparison between the three systems is carried out to examine the performance of syntactic reordering as well as the usefulness of functional words for pattern filtering.

The rest of this paper is organized as follows: in section 2 we describe the extraction process of syntactic reordering patterns, including the lattice scoring approach and the extraction procedures. Then section 3 presents the filtering process used to obtain patterns with functional words. After that, section 4 shows the generation of word lattices with patterns, and experimental setup and results included related discussion are presented in section 5. Finally, we give our conclusion and avenues for future work in section 6.

2 Syntactic reordering patterns extraction

Instead of top-down approaches such as (Wang et al., 2007a; Chang et al., 2009a), we use a bottom-up approach similar to (Xia et al., 2004; Crego et al., 2007) to extract syntactic reordering patterns from non-monotonic phrase alignments and source-side parse trees. The following steps are carried out to extract syntactic reordering patterns: 1) the lattice scoring approach proposed in (Jiang et al., 2010) is used to obtain phrase alignments from the training corpus; 2) reordering regions from the non-monotonic phrase alignments are used to identify minimum treelets for pattern extraction; and 3) the treelets are transformed into syntactic reordering patterns which are then weighted by their occurrences in the training corpus. Details of each of these steps are presented in the rest of this section.

2.1 Lattice scoring for phrase alignments

The lattice scoring approach is proposed in (Jiang et al., 2010) for the SMT data cleaning task.

To clean the training corpus, word alignments are used to obtain approximate decoding results, which are then used to calculate BLEU (Papineni et al., 2002) scores to filter out low-scoring sentences pairs. The following steps are taken in the lattice scoring approach: 1) train an initial PBSMT model; 2) collect anchor pairs containing source and target phrase positions from word alignments generated in the training phase; 3) build source-side lattices from the anchor pairs and the translation model; 4) search on the source-side lattices to obtain approximate decoding results; 5) calculate BLEU scores for the purpose of data cleaning.

Note that the source-side lattices in step 3 come from anchor pairs, so each edge in the lattices contain both the source and target phrase positions. Thus the outputs of step 4 contain phrase alignments on the training corpus. These phrase alignments are used to identify non-monotonic areas for the extraction of reordering patterns.

2.2 Reordering patterns

Non-monotonic regions of the phrase alignments are examined as potential source-side reorderings. By taking a bottom-up approach, the reordering regions are identified and mapped to minimum treelets on the source parse trees. After that, syntactic reordering patterns are transformed from these minimum treelets.

In this paper, reordering regions A and B indicating swapping operations on the source side are only considered as potential source-side reorderings. Thus, given reordering regions AB, this implies (1):

$$AB \Rightarrow BA \quad (1)$$

on the source-side word sequences. Referring to the phrase alignment extraction in the last section, each non-monotonic phrase alignment produces one reordering region. Furthermore, for each reordering region identified, all of its sub-areas indicating non-monotonic alignments are also attempted to produce more reordering regions.

To represent the reordering region using syntactic structure, given the extracted reordering regions AB, the following steps are taken to map them onto the source-side parse trees, and to generate corresponding patterns:

1. Generate a parse tree for each of the source sentences. The Berkeley parser (Petrov, 2006) is used in this paper. To obtain simpler tree structures, right-binarization is performed on the parse trees, while tags generated from binarization are not distinguished from the original ones (e.g. $@VP$ and VP are the same).

2. Map reordering regions AB onto the parse trees. Denote N_A as the set of leaf nodes in region A and N_B for region B. The mapping is carried out on the parse tree to find a **minimum** treelet T, which satisfies the following two criteria: 1) there must exist a path from each node in $N_A \cup N_B$ to the root node of T; 2) each leaf node of T can only be the ancestor of nodes in N_A or N_B (or none of them).

3. Traverse T in pre-order to obtain syntactic reordering pattern P. Label all the leaf nodes of T with A or B as reorder options, which indicate that the descendants of nodes with label A are supposed to be swapped with those with label B.

Instead of using *subtrees*, we use *treelets* to refer the located parse tree substructures, since treelets do not necessarily go down to leaf nodes.

Since phrase alignments cannot always be perfectly matched with parse trees, we also expand AB to the right and/or the left side with a limited number of words to find a minimum treelet. In this situation, a minimum number of ancestors of expanded tree nodes are kept in T but they are assigned the same labels as those from which they have been expanded. In this case, the expanded tree nodes are considered as the context nodes of syntactic reordering patterns.

Figure 1 illustrates the extraction process. Note the symbol @ indicates the right-binarization symbols (e.g. $@VP$ in the figure). In the figure, tree T (surrounded by dashed lines) is the minimum treelet mapped from the reordering region AB. Leaf node NP is labeled by A, VP is labeled by B, and the context node P is also labeled by A. Leaf nodes labeled A or B are collected into node sequences L_A or L_B to indicate the reordering op-

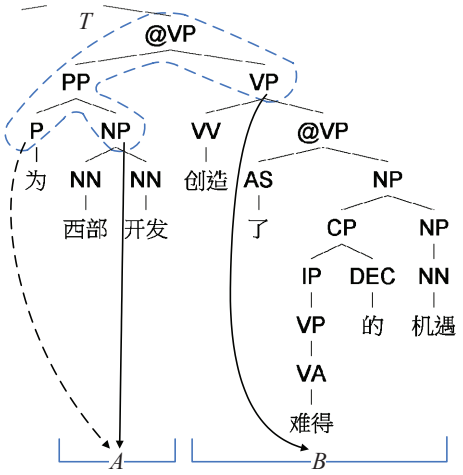


Figure 1: Reordering pattern extraction

erations. Thus the syntactic reordering pattern P is obtained from T as in (2):

$$P = \{VP\ (PP\ (P\ NP)\ VP)|O = \{L_A, L_B\}\} \quad (2)$$

where the first part of P is the VP with its tree structure, and the second part O indicates the reordering scheme, which implies that source words corresponding with descendants of L_A are supposed to be swapped with those of L_B.

2.3 Pattern weights estimation

We use p_{reo} to represent the chance of reordering when a treelet is located by a pattern on the parse tree. It is estimated by the number of reorderings for each of the occurrences of the pattern as in (3):

$$p_{reo}(P) = \frac{count\{reorderings\ \ of\ \ P\}}{count\{observation\ \ of\ \ P\}} \quad (3)$$

By contrast, one syntactic pattern P usually contains several reordering schemes (specified in formula (2)), each of them weighted as in (4):

$$w(O, P) = \frac{count\{reorderings\ \ of\ \ O\ \ in\ \ P\}}{count\{reorderings\ \ of\ \ P\}} \quad (4)$$

Generally, a syntactic reordering pattern is expressed as in (5):

$$P = \{tree \mid p_{reo} \mid O_1, w_1, \cdots, O_n, w_n\} \quad (5)$$

where $tree$ is the tree structures of the pattern, p_{reo} is the reordering probability, O_i and w_i are the reordering schemes and weights ($1 \leq i \leq n$).

3 Patterns with functional words

Some of the patterns extracted may not benefit the final system since the extraction process is controlled by phrase alignments rather than syntactic knowledge. Inspired by the study of *DE* constructions (Chang et al., 2009a; Du & Way, 2010), we assume that syntactic reorderings are indicated by functional words for the Chinese–English task. To incorporate the knowledge of functional words into the extracted patterns, instead of directly specifying the syntactic structure from the linguistic aspects, we use functional word tags to filter the extracted patterns. In this case, we assume that all patterns containing functional words tend to produce meaningful syntactic reorderings. Thus the filtered patterns carry the reordering information from the phrase alignments as well as the linguistic knowledge. Thus the noise produced in phrase alignments and the size of pattern set can be reduced, so that the speed and the performance of the system can be improved.

The functional word tags used in this paper are shown in Table 1, which come from (Xue, 2005). We choose them as functional words because normally they imply word reorders between Chinese and English sentence pairs.

Tag	Description
BA	*ba*-construction
DEC	*de* (1^{st} kind) in a relative-clause
DEG	associative *de* (1^{st} kind)
DER	*de* (2^{nd} kind) in V-de const. & V-de-R
DEV	*de* (3^{rd} kind) before VP
LB	*bei* in long *bei*-construction
P	preposition excluding *bei* and *ba*
SB	*bei* in short *bei*-construction

Table 1: Syntactic reordering tags for functional words

Note that there are three kinds of *de*-constructions, but only the first kind is the *DE* construction in (Chang et al., 2009a; Du & Way, 2010). After the filtering process, both the unfiltered pattern set and the filtered one are used to build different syntactic reordering PBSMT systems for comparison purpose.

4 Word lattice construction

Both the devset and testset are transformed into word lattices by the extracted patterns to incorporate potential reorderings. Figure 2 illustrates this process: treelet T' is matched with a pattern, then its leaf nodes $\{a_1, \cdots a_m\} \in L_A$ (spanning $\{w_1, \cdots, w_p\}$) are swapped with leaf nodes $\{b_1, \cdots, b_n\} \in L_B$ (spanning $\{v_1, \cdots, v_q\}$) on the generated paths in the word lattice.

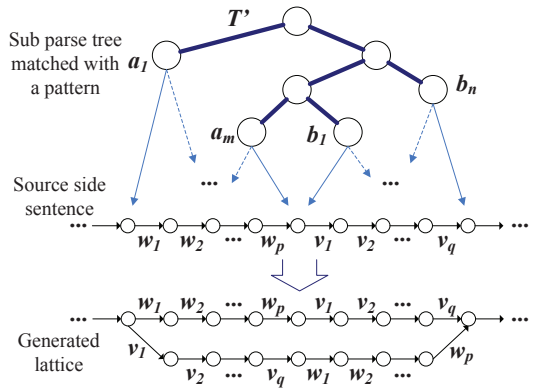


Figure 2: Incorporating potential reorderings into lattices

We sort the matched patterns by p_{reo} in formula (5), and only apply a pre-defined number of reorderings for each sentence. For each lattice node, if we denote E_0 as the edge from the original sentence, while patterns $\{P_1, \cdots, P_i, \cdots, P_k\}$ are applied to this node, then E_0 is weighted as in (6):

$$w(E_0) = \alpha + \sum_{i=1}^{k} \{\frac{(1-\alpha)}{k} * \{1 - p_{reo}(P_i)\}\} \quad (6)$$

where $p_{reo}(P_i)$ is the pattern weight in formula (3), and α is the base probability to avoid E_0 being equal to zero. Suppose $\{E_s, \cdots, E_{s+r-1}\}$ are generated by r reordering schemes of P_i, then E_j is weighted as in (7):

$$w(E_j) = \frac{(1-\alpha)}{k} * p_{reo}(P_i) * \frac{w_{s-j+1}(P_i)}{\sum_{t=1}^{r} w_t(P_i)} \quad (7)$$

where $w_t(P_i)$ is the reordering scheme in formula (5), and $s <= j < s + r$. Reordering patterns with the same root lattice node share equal probabilities in formula (6) and (7).

5 Experiments and results

We conducted our experiments on a medium-sized corpus FBIS (a multilingual paragraph-aligned corpus with LDC resource number LDC2003E14) for the Chinese–English SMT task. The Champollion aligner (Ma, 2006) is utilized to perform sentence alignment. A total number of 256,911 sentence pairs are obtained, while 2,000 pairs for devset and 2,000 pairs for testset are randomly selected, which we call FBIS set. The rest of the data is used as the training corpus.

The baseline system is Moses (Koehn et al., 2007), and GIZA++[1] is used to perform word alignment. Minimum error rate training (MERT) (Och, 2003) is carried out for tuning. A 5-gram language model built via SRILM[2] is used for all the experiments in this paper.

Experiments results are reported on two different sets: the FBIS set and the NIST set. For the NIST set, the NIST 2005 testset (1,082 sentences) is used as the devset, and the NIST 2008 testset (1,357 sentences) is used as the testset. The FBIS set contains only one reference translation for both devset and testset, while NIST set has four references.

5.1 Pattern extraction and filtering with functional words

The lattice scoring approach is carried out with the same baseline system as specified above to produce the phrase alignments. The initial PB-SMT system in the lattice scoring approach is tuned with the FBIS devset to obtain the weights. As specified in section 2.1, phrase alignments are generated in the step 4 of the lattice scoring approach.

From the generated phrase alignments and source-side parse trees of the training corpus, we obtain 48,285 syntactic reordering patterns (57,861 reordering schemes) with an average number of 11.02 non-terminals. For computational efficiency, any patterns with number of non-terminal less than 3 and more than 9 are pruned. This procedure leaves 18,169 syntactic reordering patterns (22,850 reordering schemes) with a aver-

[1] http://fjoch.com/GIZA++.html
[2] http://www.speech.sri.com/projects/srilm/

age number of 7.6 non-terminals. This pattern set is used to built the syntactic reordering PBSMT system without pattern filtering, which here after we call the 'unfiltered system'.

Using the tags specified in Table 1, the extracted syntactic reordering patterns without functional words are filtered out, while only 6,926 syntactic reordering patterns (with 9,572 reordering schemes) are retained. Thus the pattern set are reduced by 61.88%, and over half of them are pruned by the functional word tags. The filtered pattern set is used to build the syntactic reordering PBSMT system with pattern filtering, which we refer as the 'filtered system'.

Type	Tag	Patterns	Percent
ba-const.	BA	222	3.20%
bei-const.	LB SB	97 96	2.79%
de-const. (1^{st})	DEC DEG	1662 2501	60.11%
de-const. (2^{nd})	DER	52	0.75%
de-const. (3^{rd})	DEV	178	2.57%
preposition excl. *ba* & *bei*	P	2591	37.41%

Table 2: Statistics on the number of patterns for each type of functional word

Statistics on the patterns with respect to functional word types are shown in Table 2. The number of patterns for each functional word in the filtered pattern set are illustrated, and percentages of functional word types are also reported. Note that some patterns contain more than one kind of functional word, so that the percentages of functional word types do not sum to one.

As demonstrated in Table 2, the first kind of *de*-construction takes up 60.11% of the filtered pattern set, and is the main type of patterns used in our experiment. This indicates that more than half of the patterns are closely related to the *DE* construction examined in (Chang et al., 2009b; Du & Way, 2010). However, the general preposition construction (excluding *bei* and *ba*) accounts for 37.41% of the filtered patterns, which implies that it is also a major source of syntactic reordering. By contrast, other constructions have much smaller amount of percentages, so have a minor impact on our experiments.

5.2 Word lattice construction

As specified in section 4, for both unfiltered and the filtered systems, both the devset and testset are converted into word lattices with the unfiltered and filtered syntactic reordering patterns respectively. To avoid a dramatic increase in size of the lattices, the following constraints are applied: for each source sentence, the maximum number of reordering schemes is 30, and the maximum span of a pattern is 30.

For the lattice construction, the base probability in (6) and (7) is set to 0.05. The two syntactic reordering PBSMT systems also incorporate the built-in reordering models (distance-based and lexical reordering) of Moses, and their weights in the log-linear model are tuned with respect to the devsets.

The effects of the pattern filtering by functional words are also reported in Table 3. For both the FBIS and NIST sets, the average number of nodes in word lattices are illustrated before and after pattern filtering. From the table, it is clear that the pattern filtering procedure dramatically reduces the input size for the PBSMT system. The reduction is up to 37.99% for the NIST testset.

Data set	Unfiltered	Filtered	Reduced
FBIS dev	183.13	131.38	28.26%
FBIS test	183.68	136.56	25.65%
NIST dev	175.78	115.89	34.07%
NIST test	149.13	92.48	37.99%

Table 3: Comparison of the average number of nodes in word lattices

5.3 Results on FBIS set

Three systems are compared on the FBIS set: the baseline PBSMT system, and the syntactic reordering systems with and without pattern filtering. Since the built-in reordering models of Moses are enabled, several values of the distortion limit (DL) parameter are chosen to validate consistency. The evaluation results on the FBIS set are shown in Table 4.

As shown in Table 4, the syntactic reordering systems with and without pattern filtering outper-

System	DL	BLEU	NIST	METE
Baseline	0	22.32	6.45	52.51
	6	23.67	6.63	54.07
	10	24.52	6.66	54.04
	12	24.57	6.69	54.31
Unfiltered	0	**23.92**	**6.60**	**54.30**
	6	**24.57**	**6.68**	**54.64**
	10	**24.98**	**6.71**	**54.67**
	12	**24.84**	6.69	**54.65**
Filtered	0	23.71	6.60	54.11
	6	**24.65**	6.68	54.61
	10	24.87	6.71	**54.84**
	12	**24.91**	**6.7**	54.51

Table 4: Results on FBIS testset (DL = distortion limit, METE=METEOR)

form the baseline system for each of the distortion limit parameters in terms of the BLEU, NIST and METEOR scores (scores in bold face). By contrast, the filtered systems has a comparable performance with the unfiltered system: for some of the distortion limits, the filtered systems even outperforms the unfiltered system (scores in bold face, e.g. BLEU and NIST for DL=12, METEOR for DL=10).

The best performance of the baseline system is obtained with distortion limit 12 (underlined); the best performance of the unfiltered system is achieved with distortion limit 10 (underlined); while for the filtered system, the best BLEU score is accomplished with distortion limit 12 (underlined), and the best NIST and METEOR scores are shown with distortion limit 10 (underlined). Thus the unfiltered system outperforms the baseline by 0.41 (1.67% relative) BLEU points, 0.02 (0.30% relative) NIST points and 0.36 (0.66% relative) METEOR points. By contrast, the filtered system outperforms the baseline by 0.34 (1.38% relative) BLEU points, 0.02 (0.30% relative) NIST points and 0.53 (0.98% relative) METEOR points.

Compared with the unfiltered system, pattern filtering with functional words degrades performance by 0.07 (0.28% relative) in term of BLEU, but improves the system by 0.17 (0.31% relative) in term of METEOR, while the two systems achieved the same best NIST score.

These results indicates that the filtered system has a comparable performance with the unfiltered one on the FBIS set, while both of them outperform the baseline.

5.4 Results on NIST set

The evaluation results on the NIST set are illustrated in Table 5.

System	DL	BLEU	NIST	METE
Baseline	0	14.43	5.75	45.03
	6	15.61	5.88	45.75
	10	15.73	5.78	45.27
	12	15.89	6.16	45.88
Unfiltered	0	**16.77**	**6.54**	**47.16**
	6	**17.25**	**6.67**	**47.65**
	10	**17.15**	**6.64**	**47.78**
	12	**16.88**	**6.56**	**47.17**
Filtered	0	**16.79**	**6.64**	**47.67**
	6	**17.55**	**6.71**	**48.06**
	10	**17.51**	**6.72**	**48.15**
	12	**17.37**	**6.72**	**48.08**

Table 5: Results on NIST testset (DL = distortion limit, METE=METEOR)

From Table 5, the unfiltered system outperforms the baseline system for each of the distortion limits in terms of the BLEU, NIST and METEOR scores (scores in bold face). By contrast, the filtered system also outperform the unfiltered system for each of the distortion limits in terms of the three evaluation methods (scores in bold face).

The best performance of the baseline system is obtained with distortion limit 12 (underlined), while the best performance of the unfiltered system is obtained with distortion limit 6 for BLEU and NIST, and 10 for METEOR (underlined). For the filtered system, the best BLEU score is shown with distortion limit 6, and the best NIST and METEOR scores are accomplished with distortion limit 10 (underlined). Thus the unfiltered system outperforms the baseline by 1.36 (8.56% relative) BLEU points, 0.51 (8.28% relative) NIST points and 1.90 (4.14% relative) METEOR points. By contrast, the filtered system outperforms the baseline by 1.66 (10.45% relative) BLEU points, 0.56 (9.52% relative) NIST points and 2.27 (4.95% relative) METEOR points.

Compared with the unfiltered system, patterns with functional words boost the performance by 0.30 (1.74% relative) in term of BLEU, 0.05 (0.75% relative) in term of NIST, and 0.37 (0.77% relative) in term of METEOR.

These results demonstrate that the pattern filtering improves the syntactic reordering system on the NIST set, while both of them significantly outperform the baseline.

5.5 Discussion

Experiments in the previous sections demonstrate that: 1) the two syntactic reordering systems improve the PBSMT system by providing potential reorderings obtained from phrase alignments and parse trees; 2) patterns with functional words play a major role in the syntactic reordering process, and filtering the patterns with functional words maintains or even improves the system performance for Chinese–English SMT task. Furthermore, as shown in the previous section, pattern filtering prunes the whole pattern set by 61.88% and also reduces the sizes of word lattices by up to 37.99%, thus the whole syntactic reordering procedure for the original inputs as well as the tuning/decoding steps are sped up dramatically, which make the proposed methods more useful in the real world, especially for online SMT systems.

From the statistics on the filtered pattern set in Table 2, we also argue that the first kind of *de*-construction and general preposition (excluding *bei* and *ba*) are the main sources of Chinese–English syntactic reordering. Previous work (Chang et al., 2009b; Du & Way, 2010) showed the advantages of dealing with the *DE* construction. In our experiments too, even though all the patterns are automatically extracted from phrase alignments, these two constructions still dominate the filtered pattern set. This result confirms the effectiveness of previous work on *DE* construction, and also highlights the importance of the general preposition construction in this task.

6 Conclusion and future work

Syntactic reordering patterns with functional words are examined in this paper. The aim is to exploit these functional words within the syntactic reordering patterns extracted from phrase alignments and parse trees. Three systems are compared: a baseline PBSMT system, a syntactic reordering system with all patterns extracted from a corpus and a syntactic reordering system with patterns filtered with functional words. Evaluation results on a medium-sized corpus showed that the two syntactic reordering systems consistently outperform the baseline system. The pattern filtering with functional words prunes 61.88% of patterns, but still maintains a comparable performance with the unfiltered one on the randomly select testset, and even obtains 1.74% relative improvement on the NIST 2008 testset.

In future work, the structures of patterns containing functional words will be investigated to obtain fine-grained analysis on such words in this task. Furthermore, experiments on larger corpora as well as on other language pairs will also be carried out to validation our method.

Acknowledgements

This research is supported by Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University. Thanks to Yanjun Ma for the sentence-aligned FBIS corpus.

References

Yaser Al-Onaizan and Kishore Papineni 2006. Distortion models for statistical machine translation. *Coling-ACL 2006: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 529-536, Sydney, Australia.

Pi-Chuan Chang, Dan Jurafsky, and Christopher D.Manning 2009a. Disambiguating DE for Chinese–English machine translation. *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 215-223, Athens, Greece.

Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009b. Discriminative reordering with Chinese grammatical features. *Proceedings of SSST-3: Third Workshop on Syntax and Structure in Statistical Translation*, pages 51-59, Boulder, CO.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. *ACL-2005: 43rd Annual meeting of the*

Association for Computational Linguistics, pages 531-540, University of Michigan, Ann Arbor, MI.

Josep M. Crego, and José B. Mariño. 2007. Syntax-enhanced N-gram-based SMT. *MT Summit XI*, pages 111-118, Copenhagen, Denmark.

Jinhua Du and Andy Way. 2010. The Impact of Source-Side Syntactic Reordering on Hierarchical Phrase-based SMT. *EAMT 2010: 14th Annual Conference of the European Association for Machine Translation*, Saint-Raphaël, France.

Jakob Elming. 2008. Syntactic reordering integrated with phrase-based SMT. *Coling 2008: 22nd International Conference on Computational Linguistics, Proceedings of the conference*, pages 209-216, Manchester, UK.

Jakob Elming, and Nizar Habash. 2009. Syntactic reordering for English-Arabic phrase-based machine translation. *Proceedings of the EACL 2009 Workhop on Computational Approaches to Semitic Languages*, pages 69-77, Athens, Greece.

Nizar Habash. 2007. Syntactic preprocessing for statistical machine translation. *MT Summit XI*, pages 215-222, Copenhagen, Denmark.

Jie Jiang, Andy Way, Julie Carson-Berndsen. 2010. Lattice Score-Based Data Cleaning For Phrase-Based Statistical Machine Translation. *EAMT 2010: 14th Annual Conference of the European Association for Machine Translation*, Saint-Raphaël, France.

Philipp Koehn, 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. *ACL 2007: proceedings of demo and poster sessions*, pp. 177-180, Prague, Czech Republic.

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. *ACL 2007: proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 720-727, Prague, Czech Republic.

Xiaoyi Ma. 2006. Champollion: A Robust Parallel Text Sentence Aligner. *LREC 2006: Fifth International Conference on Language Resources and Evaluation*, pp.489-492, Genova, Italy.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. *ACL-2003: 41st Annual meeting of the Association for Computational Linguistics*, pp. 160-167, Sapporo, Japan.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method For Automatic Evaluation of Machine Translation. *ACL-2002: 40th Annual meeting of the Association for Computational Linguistics*, pp.311-318, Philadelphia, PA.

Slav Petrov, Leon Barrett, Romain Thibaux and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. *Coling-ACL 2006: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433-440, Sydney, Australia.

Felipe Sánchez-Martínez and Andy Way. 2009. Marker-based filtering of bilingual phrase pairs for SMT. *EAMT-2009: Proceedings of the 13th Annual Conference of the European Association for Machine Translation*, pages 144-151, Barcelona, Spain.

Chao Wang, Michael Collins, and Philipp Koehn. 2007a. Chinese syntactic reordering for statistical machine translation. *EMNLP-CoNLL-2007: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 737-745, Prague, Czech Republic.

Fei Xia, and Michael McCord 2004. Improving a statistical MT system with automatically learned rewrite patterns. *Coling 2004: 20th International Conference on Computational Linguistics*, pages 508-514, University of Geneva, Switzerland.

Nianwen Xue, Fei Xia, Fu-dong Chiou, and Martha Palmer. 2005. The Penn Chinese TreeBank: Phrase structure annotation of a large corpus. *Natural Language Engineering*, 11(2), pages 207-238.

Richard Zens, Franz Josef Och, and Hermann Ney. 2002. Phrase-based statistical machine translation. *Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP*, pages 333-341, Suntec, Singapore.

Yuqi Zhang, Richard Zens, and Hermann Ney 2007a. Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. *SSST, NAACL-HLT-2007 AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 1-8, Rochester, NY.

Yuqi Zhang, Richard Zens, and Hermann Ney 2007b. Improved chunk-level reordering for statistical machine translation. *IWSLT 2007: International Workshop on Spoken Language Translation*, pages 21-28, Trento, Italy.

Syntactic Constraints on Phrase Extraction for Phrase-Based Machine Translation

Hailong Cao, Andrew Finch and Eiichiro Sumita
Language Translation Group, MASTAR Project
National Institute of Information and Communications Technology
{hlcao,andrew.finch,eiichiro.sumita }@nict.go.jp

Abstract

A typical phrase-based machine translation (PBMT) system uses phrase pairs extracted from word-aligned parallel corpora. All phrase pairs that are consistent with word alignments are collected. The resulting phrase table is very large and includes many non-syntactic phrases which may not be necessary. We propose to filter the phrase table based on source language syntactic constraints. Rather than filter out all non-syntactic phrases, we only apply syntactic constraints when there is phrase segmentation ambiguity arising from unaligned words. Our method is very simple and yields a 24.38% phrase pair reduction and a 0.52 BLEU point improvement when compared to a baseline PBMT system with full-size tables.

1 Introduction

Both PBMT models (Koehn et al., 2003; Chiang, 2005) and syntax-based machine translation models (Yamada et al., 2000; Quirk et al., 2005; Galley et al., 2006; Liu et al., 2006; Marcu et al., 2006; and numerous others) are the state-of-the-art statistical machine translation (SMT) methods. Over the last several years, an increasing amount of work has been done to combine the advantages of the two approaches. DeNeefe et al. (2007) made a quantitative comparison of the phrase pairs that each model has to work with and found it is useful to improve the phrasal coverage of their string-to-tree model. Liu et al. (2007) proposed forest-to-string rules to capture the non-syntactic phrases in their tree-to-string model. Zhang et al. (2008) proposed a tree sequence based tree-to-tree model which can describe non-syntactic phrases with syntactic structure information.

The converse of the above methods is to incorporate syntactic information into the PBMT model. Zollmann and Venugopal (2006) started with a complete set of phrases as extracted by traditional PBMT heuristics, and then annotated the target side of each phrasal entry with the label of the constituent node in the target-side parse tree that subsumes the span. Marton and Resnik (2008) and Cherry (2008) imposed syntactic constraints on the PBMT system by making use of prior linguistic knowledge in the form of syntax analysis. In their PBMT decoders, a candidate translation gets an extra credit if it respects the source side syntactic parse tree but may incur a cost if it violates a constituent boundary. Xiong et al. (2009) proposed a syntax-driven bracketing model to predict whether a phrase (a sequence of contiguous words) is bracketable or not using rich syntactic constraints.

In this paper, we try to utilize syntactic knowledge to constrain the phrase extraction from word-based alignments for PBMT system. Rather than filter out all non-syntactic phrases, we only apply syntactic constraints when there is phrase segmentation ambiguity arising from unaligned words. Our method is very simple and yields a 24.38% phrase pair reduction and a 0.52 BLEU point improvement when compared to the baseline PBMT system with full-size tables.

2 Extracting Phrase Pairs from Word-based Alignments

In this section, we briefly review a simple and effective phrase pair extraction algorithm upon which this work builds.

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 28–33,
COLING 2010, Beijing, August 2010.

The basic translation unit of a PBMT model is the phrase pair, which consists of a sequence of source words, a sequence of target words and a vector of feature values which represents this pair's contribution to the translation model. In typical PBMT systems such as MOSES (Koehn, 2007), phrase pairs are extracted from word-aligned parallel corpora. Figure 1 shows the form of training example.

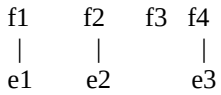


Figure 1: An example parallel sentence pair and word alignment

Since there is no phrase segmentation information in the word-aligned sentence pair, in practice all pairs of "source word sequence ||| target word sequence" that are consistent with word alignments are collected. The words in a legal phrase pair are only aligned to each other, and not to words outside (Och et al., 1999). For example, given a sentence pair and its word alignments shown in Figure1, the following nine phrase pairs will be extracted:

Source phrase ||| Target phrase
f1 ||| e1
f2 ||| e2
f4 ||| e3
f1 f2 ||| e1 e2
f2 f3 ||| e2
f3 f4 ||| e3
f1 f2 f3 ||| e1 e2
f2 f3 f4 ||| e2 e3
f1 f2 f3 f4 ||| e1 e2 e3

Table 1: Phrase pairs extracted from the example in Figure 1

Note that neither the source phrase nor the target phrase can be empty. So "f3 ||| EMPTY" is not a legal phrase pair.

Phrase pairs are extracted over the entire training corpus. Given all the collected phrase pairs, we can estimate the phrase translation probability distribution by relative frequency. The collected phrase pairs will also be used to build the lexicalized reordering model. For more details of the lexicalized reordering model, please refer to Tillmann and Zhang (2005) and section 2.7.2 of the MOSES's manual[1].

The main problem of such a phrase pair extraction procedure is the resulting phrase translation table is very large, especially when a large quantity of parallel data is available. This is not desirable in real application where speed and memory consumption are often critical concerns. In addition, some phrase translation pairs are generated from training data errors and word alignment noise. Therefore, we need to filter the phrase table in an appropriate way for both efficiency and translation quality (Johnson et al., 2007; Yang and Zheng, 2009).

3 Syntactic Constraints on Phrase Pair Extraction

We can divide all the possible phrases into two types: syntactic phrases and non-syntactic phrases. A "syntactic phrase" is defined as a word sequence that is covered by a single subtree in a syntactic parse tree (Imamura, 2002). Intuitively, we would think syntactic phrases are much more reliable while the non-syntactic phrases are useless. However, (Koehn et al., 2003) showed that restricting phrasal translation to only syntactic phrases yields poor translation performance – the ability to translate non-syntactic phrases (such as "there are", "note that", and "according to") turns out to be critical and pervasive.

(Koehn et al., 2003) uses syntactic constraints from both the source and target languages, and over 80% of all phrase pairs are eliminated. In this section, we try to use syntactic knowledge in a less restrictive way.

Firstly, instead of using syntactic restriction on both source phrases and target phrases, we only apply syntactic restriction to the source language side.

Secondly, we only apply syntactic restriction to the source phrase whose first or last word is unaligned.

For example, given a parse tree illustrated in Figure 2, we will filter out the phrase pair "f2 f3 ||| e2" since the source phrase "f2 f3" is a non-syntactic phrase and its last word "f3" is not

[1] http://www.statmt.org/moses/

aligned to any target word. The phrase pair "f1 f2 f3 ||| e1 e2" will also be eliminated for the same reason. But we do keep phrase pairs such as "f1 f2 ||| e1 e2" even if its source phrase "f1 f2" is a non-syntactic phrase. Also, we keep "f3 f4 ||| e3" since "f3 f4" is a syntactic phrase. Table 2 shows the completed set of phrase pairs that are extracted with our constraint-based method.

Source phrase ||| Target phrase
f1 ||| e1
f2 ||| e2
f4 ||| e3
f1 f2 ||| e1 e2
f3 f4 ||| e3
f2 f3 f4 ||| e2 e3
f1 f2 f3 f4 ||| e1 e2 e3

Table 2: Phrase pairs extracted from the example in Figure 2

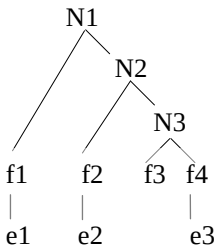


Figure 2: An example parse tree and word-based alignments

The state-of-the-art alignment tool such as GIZA++[2] can not always find alignments for every word in the sentence pair. The possible reasons could be: its frequency is too low, noisy data, auxiliary words or function words which have no obvious correspondence in the opposite language.

In the automatically aligned parallel corpus, unaligned words are frequent enough to be noticeable (see section 4.1 in this paper). How to decide the translation of unaligned word is left to the phrase extraction algorithm. An unaligned source word should be translated together with the words on the right of it or the words on the left of it. The existing algorithm considers both of the two directions. So both "f2 f3 ||| e2" and "f3 f4 ||| e3" are extracted. However, it is unlikely that "f3" can be translated into both "e2" and "e3". So our algorithm uses prior syntactic knowledge to keep "f3 f4 ||| e3" and exclude "f2 f3 ||| e2".

4 Experiments

Our SMT system is based on a fairly typical phrase-based model (Finch and Sumita, 2008). For the training of our SMT model, we use a modified training toolkit adapted from the MOSES decoder. Our decoder can operate on the same principles as the MOSES decoder. Minimum error rate training (MERT) with respect to BLEU score is used to tune the decoder's parameters, and it is performed using the standard technique of Och (2003). A lexicalized reordering model was built by using the "msd-bidirectional-fe" configuration in our experiments.

The translation model was created from the FBIS parallel corpus. We used a 5-gram language model trained with modified Kneser-Ney smoothing. The language model was trained on the target side of the FBIS corpus and the Xinhua news in the GIGAWORD corpus. The development and test sets are from the NIST MT08 evaluation campaign. Table 3 shows the statistics of the corpora used in our experiments.

Data	Sentences	Chinese words	English words
Training set	221,994	6,251,554	8,065,629
Development set	1,664	38,779	46,387
Test set	1,357	32,377	42,444
GIGAWORD	19,049,757	-	306,221,306

Table 3: Corpora statistics

The Chinese sentences are segmented, POS tagged and parsed by the tools described in Kruengkrai et al. (2009) and Cao et al. (2007), both of which are trained on the Penn Chinese Treebank 6.0.

[2] http://fjoch.com/GIZA++.html

4.1 Experiments on Word Alignments

We use GIZA++ to align the sentences in both the Chinese-English and English-Chinese directions. Then we combine the alignments using the standard "grow-diag-final-and" procedure provided with MOSES.

In the combined word alignments, 614,369 or 9.82% of the Chinese words are unaligned. Table 4 shows the top 10 most frequently unaligned words. Basically, these words are auxiliary words or function words whose usage is very flexible. So it would be difficult to automatically align them to the target words.

Unaligned word	Frequency
的	77776
,	29051
在	9414
一	8768
中	8543
个	7471
是	7365
上	6155
了	5945
不	5450

Table 4: Frequently unaligned words from the training corpus

4.2 Experiments on Chinese-English SMT

In order to confirm that it is advantageous to apply appropriate syntactic constraints on phrase extraction, we performed three translation experiments by using different ways of phrase extraction.

In the first experiment, we used the method introduced in Section 2 to extract all possible phrase translation pairs without using any constraints arising from knowledge of syntax.

The second experiment used source language syntactic constraints to filter out all non-syntactic phrases during phrase pair extraction.

The third experiment used source language syntactic constraints to filter out only non-syntactic phrases whose first or last source word was unaligned.

With the exception of the above differences in phrase translation pair extraction, all the other settings were the identical in the three experiments. Table 5 summarizes the SMT performance. The evaluation metric is case-sensitive BLEU-4 (Papineni et al., 2002) which estimates the accuracy of translation output with respect to a set of reference translations.

Syntactic Constraints	Number of distinct phrase pairs	BLEU
None	14,195,686	17.26
Full constraint	4,855,108	16.51
Selectively constraint	10,733,731	17.78

Table 5: Comparison of different constraints on phrase pair extraction by translation quality

As shown in the table, it is harmful to fully apply syntactic constraints on phrase extraction, even just on the source language side. This is consistent with the observation of (Koehn et al., 2003) who applied both source and target constraints in German to English translation experiments.

Clearly, we obtained the best performance if we use source language syntactic constraints only on phrases whose first or last source word is unaligned. In addition, we reduced the number of distinct phrase pairs by 24.38% over the baseline full-size phrase table.

The results in table 5 show that while some non-syntactic phrases are very important to maintain the performance of a PBMT system, not all of them are necessary. We can achieve better performance and a smaller phrase table by applying syntactic constraints when there is phrase segmentation ambiguity arising from unaligned words.

5 Related Work

To some extent, our idea is similar to Ma et al. (2008), who used an anchor word alignment model to find a set of high-precision anchor links and then aligned the remaining words relying on dependency information invoked by the acquired anchor links. The similarity is that both Ma et al. (2008) and this work utilize structure information to find appropriate translations for words which are difficult to align. The differ-

ence is that they used dependency information in the word alignment stage while our method uses syntactic information during the phrase pair extraction stage. There are also many works which leverage syntax information to improve word alignments (e.g., Cherry and Lin, 2006; DeNero and Klein, 2007; Fossum et al., 2008; Hermjakob, 2009).

Johnson et al., (2007) presented a technique for pruning the phrase table in a PBMT system using Fisher's exact test. They compute the significance value of each phrase pair and prune the table by deleting phrase pairs with significance values smaller than a certain threshold. Yang and Zheng (2008) extended the work in Johnson et al., (2007) to a hierarchical PBMT model, which is built on synchronous context free grammars (SCFG). Tomeh et al., (2009) described an approach for filtering phrase tables in a statistical machine translation system, which relies on a statistical independence measure called *Noise*, first introduced in (Moore, 2004). The difference between the above research and this work is they took advantage of some statistical measures while we use syntactic knowledge to filter phrase tables.

6 Conclusion and Future Work

Phrase pair extraction plays a very important role on the performance of PBMT systems. We utilize syntactic knowledge to constrain the phrase extraction from word-based alignments for a PBMT system. Rather than filter out all non-syntactic phrases, we only filter out non-syntactic phrases whose first or last source word is unaligned. Our method is very simple and yields a 24.38% phrase pair reduction and a 0.52 BLEU point improvement when compared to the baseline PBMT system with full-size tables.

In the future work, we will use other language pairs to test our phrase extraction method so that we can discover whether or not it is language independent.

References

Robert C. Moore. 2004. On log-likelihood-ratios and the significance of rare events. In *EMNLP*.

Hailong Cao, Yujie Zhang and Hitoshi Isahara. Empirical study on parsing Chinese based on Collins' model. 2007. In *PACLING*.

Colin Cherry and Dekang Lin. 2006. Soft syntactic constraints for word alignment through discriminative training. In *ACL*.

Colin Cherry. 2008. Cohesive phrase-Based decoding for statistical machine translation. In *ACL-HLT*.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *ACL*.

Steve DeNeefe, Kevin Knight, Wei Wang, and Daniel Marcu. 2007. What can syntax-based MT learn from phrase-based MT? In *EMNLP-CoNLL*.

John DeNero and Dan Klein. 2007. Tailoring word alignments to syntactic machine translation. In *ACL*.

Andrew Finch and Eiichiro Sumita. 2008. Dynamic model interpolation for statistical machine translation. In *SMT Workshop*.

Victoria Fossum, Kevin Knight and Steven Abney. 2008. Using syntax to improve word alignment precision for syntax-based machine translation. In *SMT Workshop, ACL*.

Michel Galley, 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 *ACL*.

Ulf Hermjakob. 2009. Improved word alignment with statistics and linguistic heuristics. In *EMNLP*.

Kenji Imamura. 2002. Application of translation knowledge acquired by hierarchical phrase alignment for pattern-based MT. In *TMI*.

Howard Johnson, Joel Martin, George Foster and Roland Kuhn. 2007. Improving translation quality by discarding most of the phrase table. In *EMNLP-CoNLL*.

Franz Josef Och, Christoph Tillmann and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *EMNLP-VLC*.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *ACL*.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL*.

Philipp Koehn, 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, Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *ACL demo and poster sessions.*

Canasai Kruengkrai, 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 *ACL-IJCNLP*.

Yang Liu, Qun Liu, Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *ACL-COLING*.

Yang Liu, Yun Huang, Qun Liu and Shouxun Lin. 2007. Forest-to-string statistical translation rules. In *ACL*.

Yanjun Ma, Sylwia Ozdowska, Yanli Sun and Andy Way. 2008. Improving word alignment using syntactic dependencies. In *SSST*.

Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *EMNLP*.

Yuval Marton and Philip Resnik. 2008. Soft syntactic constraints for hierarchical phrased-based translation. In *ACL-HLT*.

Kishore Papineni, Salim Roukos, Todd Ward and WeiJing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *ACL*.

Chris Quirk and Arul Menezes and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *ACL*.

Christoph Tillmann and Tong Zhang. 2005. A localized prediction model for statistical machine translation. In *ACL*.

Nadi Tomeh, Nicola Cancedda and Marc Dymetman. 2009. Complexity-based phrase-table filtering for statistical machine translation. In *MT Summit*.

Deyi Xiong, Min Zhang, Aiti Aw and Haizhou Li. 2009. A syntax-driven bracketing model for phrase-based translation. In *ACL-IJCNLP*.

Kenji Yamada and Kevin Knight. 2000. A syntax-based statistical translation model. In *ACL*.

Mei Yang and Jing Zheng. 2009. Toward smaller, faster, and better hierarchical phrase-based SMT. In *ACL*.

Min Zhang, Hongfei Jiang, Aiti Aw, Chew Lim Tan and Sheng Li. 2008. A tree sequence alignment-based tree-to-tree translation model. In *ACL- HLT*.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *SMT Workshop, HLT-NAACL*.

Phrase Based Decoding using a Discriminative Model

Prasanth Kolachina
LTRC, IIIT-Hyderabad
{prasanth_k}@research.iiit.ac.in

Sriram Venkatapathy
LTRC, IIIT-Hyderabad
{sriram}@research.iiit.ac.in

Srinivas Bangalore
AT&T Labs-Research, NY
{srini}@research.att.com

Sudheer Kolachina
LTRC, IIIT-Hyderabad
{sudheer.kpg08}@research.iiit.ac.in

Avinesh PVS
LTRC, IIIT-Hyderabad
{avinesh}@research.iiit.ac.in

Abstract

In this paper, we present an approach to statistical machine translation that combines the power of a discriminative model (for training a model for Machine Translation), and the standard beam-search based decoding technique (for the translation of an input sentence). A discriminative approach for learning lexical selection and reordering utilizes a large set of feature functions (thereby providing the power to incorporate greater contextual and linguistic information), which leads to an effective training of these models. This model is then used by the standard state-of-art Moses decoder (Koehn et al., 2007) for the translation of an input sentence.

We conducted our experiments on Spanish-English language pair. We used maximum entropy model in our experiments. We show that the performance of our approach (using simple lexical features) is comparable to that of the state-of-art statistical MT system (Koehn et al., 2007). When additional syntactic features (POS tags in this paper) are used, there is a boost in the performance which is likely to improve when richer syntactic features are incorporated in the model.

1 Introduction

The popular approaches to machine translation use the generative IBM models for training (Brown et al., 1993; Och et al., 1999). The parameters for these models are learnt using the standard EM Algorithm. The parameters used in these models are extremely restrictive, that is, a simple, small and closed set of feature functions is used to represent the translation process. Also, these feature functions are local and are word based. In spite of these limitations, these models perform very well for the task of word-alignment because of the restricted search space. However, they perform poorly during decoding (or translation) because of their limitations in the context of a much larger search space.

To handle the contextual information, phrase-based models were introduced (Koehn et al., 2003). The phrase-based models use the word alignment information from the IBM models and train source-target phrase pairs for lexical selection (phrase-table) and distortions of source phrases (reordering-table). These models are still relatively local, as the target phrases are tightly associated with their corresponding source phrases. In contrast to a phrase-based model, a discriminative model has the power to integrate much richer contextual information into the training model. Contextual information is extremely useful in making lexical selections of higher quality, as illustrated by the models for Global Lexical Selection (Bangalore et al., 2007; Venkatapathy and

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 34–42,
COLING 2010, Beijing, August 2010.

Bangalore, 2009).

However, the limitation of global lexical selection models has been sentence construction. In global lexical selection models, lattice construction and scoring (LCS) is used for the purpose of sentence construction (Bangalore et al., 2007; Venkatapathy and Bangalore, 2009). In our work, we address this limitation of global lexical selection models by using an existing state-of-art decoder (Koehn et al., 2007) for the purpose of sentence construction. The translation model used by this decoder is derived from a discriminative model, instead of the usual phrase-table and reordering-table construction algorithms. This allows us to use the effectiveness of an existing phrase-based decoder while retaining the advantages of the discriminative model. In this paper, we compare the sentence construction accuracies of lattice construction and scoring approach (see section 4.1 for LCS Decoding) and the phrase-based decoding approach (see section 4.2).

Another advantage of using a discriminative approach to construct the phrase table and the reordering table is the flexibility it provides to incorporate linguistic knowledge in the form of additional feature functions. In the past, factored phrase-based approaches for Machine Translation have allowed the use of linguistic feature functions. But, they are still bound by the locality of context, and definition of a fixed structure of dependencies between the factors (Koehn and Hoang, 2007). Furthermore, factored phrase-based approaches place constraints both on the type and number of factors that can be incorporated into the training. In this paper, though we do not extensively test this aspect, we show that using syntactic feature functions does improve the performance of our approach, which is likely to improve when much richer syntactic feature functions (such as information about the parse structure) are incorporated in the model.

As the training model in a standard phrase-based system is relatively impoverished with respect to contextual/linguistic information, integration of the discriminative model in the form of phrase-table and reordering-table with the phrase-based decoder is highly desirable. We propose to do this by defining sentence specific tables. For example, given a source sentence s, the phrase-table contains all the possible phrase-pairs conditioned on the context of the source sentence s.

In this paper, the key contributions are,

1. We combine a discriminative training model with a phrase-based decoder. We obtained comparable results with the state-of-art phrase-based decoder.

2. We evaluate the performance of the lattice construction and scoring (LCS) approach to decoding. We observed that even though the lexical accuracy obtained using LCS is high, the performance in terms of sentence construction is low when compared to phrase-based decoder.

3. We show that the incorporation of syntactic information (POS tags) in our discriminative model boosts the performance of translation. In future, we plan to use richer syntactic feature functions (which the discriminative approach allows us to incorporate) to evaluate the approach.

The paper is organized in the following sections. Section 2 presents the related work. In section 3, we describe the training of our model. In section 4, we present the decoding approaches (both LCS and phrase-based decoder). We describe the data used in our experiments in section 5. Section 6 consists of the experiments and results. Finally we conclude the paper in section 7.

2 Related Work

In this section, we present approaches that are directly related to our approach. In Direct Translation Model (DTM) proposed for statistical machine translation by (Papineni et al., 1998; Och and Ney, 2002), the authors present a discriminative set-up for natural language understanding (and MT). They use a slightly modified equation (in comparison to IBM models) as shown in equation 1. In equation 1, they consider the translation model from $f \rightarrow e$ ($p(e|f)$), instead of the theoretically sound (after the application of Bayes' rule), $e \rightarrow f$ ($p(f|e)$) and use grammatical features such as the presence of equal number of

verbs forms etc.

$$\hat{e} = \arg\max_{e} p_{TM}(e|f) * p_{LM}(e) \quad (1)$$

In their model, they use generic feature functions such as language model, cooccurence features such as presence of a lexical relationship in the lexicon. Their search algorithm limited the use of complex features.

Direct Translation Model 2 (DTM2) (Ittycheriah and Roukos, 2007) expresses the phrase-based translation task in a unified log-linear probabilistic framework consisting of three components:

1. a prior conditional distribution P_0
2. a number of feature functions $\Phi_i()$ that capture the effects of translation and language model
3. the weights of the features λ_i that are estimated using MaxEnt training (Berger et al., 1996) as shown in equation 2.

$$Pr(e|f) = \frac{P_0(e,j|f)}{Z} exp \sum_i \lambda_i \Phi_i(e,j,f) \quad (2)$$

In the above equation, j is the skip reordering factor for the phrase pair captured by $\Phi_i()$ and represents the jump from the previous source word. Z represents the per source sentence normalization term (Hassan et al., 2009). While a uniform prior on the set of futures results in a *maximum entropy* model, choosing other priors output a *minimum divergence* models. Normalized phrase count has been used as the prior P_0 in the DTM2 model.
The following decision rule is used to obtain optimal translation.

$$\begin{aligned} \hat{e} &= \arg\max_{e} Pr(e|f) \\ &= \arg\max_{e} \sum_{m=1}^{M} \lambda_m \Phi_m(f,e) \end{aligned} \quad (3)$$

The DTM2 model differs from other phrase-based SMT models in that it avoids the redundancy present in other systems by extracting from a word aligned parallel corpora a set of minimal phrases such that no two phrases overlap with each other (Hassan et al., 2009).

The decoding strategy in DTM2 (Ittycheriah and Roukos, 2007) is similar to a phrase-based decoder except that the score of a particular translation block is obtained from the maximum entropy model using the set of feature functions. In our approach, instead of providing the complete scoring function ourselves, we compute the parameters needed by a phrase based decoder, which in turn uses these parameters appropriately. In comparison with the DTM2, we also use minimal non-overlapping blocks as the entries in the phrase table that we generate.

Xiong et al. (2006) present a phrase reordering model under the ITG constraint using a maximum entropy framework. They model the reordering problem as a two-class classification problem, the classes being *straight* and *inverted*. The model is used to merge the phrases obtained from translating the segments in a source sentence. The decoder used is a hierarchical decoder motivated from the CYK parsing algorithm employing a beam search algorithm. The maximum entropy model is presented with features extracted from the blocks being merged and probabilities are estimated using the log-linear equation shown in (4). The work in addition to lexical features and collocational features, uses an additional metric called the information gain ratio (IGR) as a feature. The authors report an improvement of 4% BLEU score over the traditional distance based distortion model upon using the lexical features alone.

$$p_\lambda(y|x) = \frac{1}{Z_\lambda(x)} exp(\sum_i \lambda_i \Phi_i(x,y)) \quad (4)$$

3 Training

The training process of our approach has two steps:

1. training the discriminative models for translation and reordering.
2. integrating the models into a phrase based decoder.

The input to our training step are the word-alignments between source and target sentences obtained using GIZA++ (implementation of IBM, HMM models).

3.1 Training discriminative models

We train two models, one to model the translation of source blocks, and the other to model the reordering of source blocks. We call the translation model a 'context dependent block translation model' for two reasons.

1. It is concerned with the translation of minimal phrasal units called blocks.

2. The context of the source block is used during its translation.

The word alignments are used to obtain the set of possible target blocks, and are added to the target vocabulary. A target block b is a sequence of n words that are paired with a sequence of m source words (Ittycheriah and Roukos, 2007). In our approach, we restrict ourselves to target blocks that are associated with only one source word. However, this constraint can be easily relaxed.

Similarly, we call the reordering model, a 'context dependent block distortion model'. For training, we use the maximum entropy software library Llama presented in (Haffner, 2006).

3.1.1 Context Dependent Block Translation Model

In this model, the goal is to predict a target block given the source word and contextual and syntactic information. Given a source word and its lexical context, the model estimates the probabilities of the presence or absence of possible target blocks (see Figure 1).

The probabilities of the candidate target blocks are obtained from the maximum entropy model. The probability p_{e_i} of a candidate target block e_i is estimated as given in equation 5

$$p_{e_i} = P(true|e_i, f_j, C) \quad (5)$$

where f_j is the source word corresponding to e_i and C is its context.

Using the maximum entropy model, binary classifiers are trained for every target block in the vocabulary. These classifiers predict if a particular target block should be present given the source word and its context. This model is similar to the global lexical selection (GLS) model described in (Bangalore et al., 2007; Venkatapathy and Bangalore, 2009) except that in GLS, the predicted target blocks are not associated with any particular source word unlike the case here.

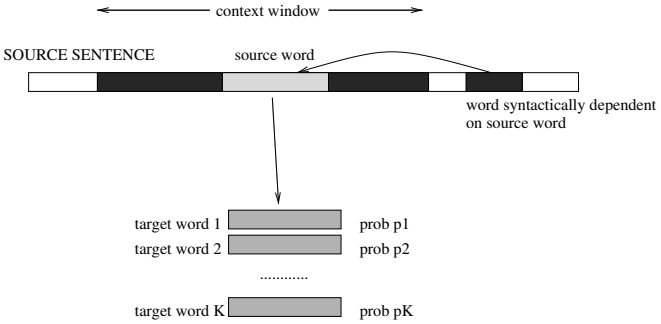


Figure 1: Word prediction model

For the set of experiments in this paper, we used a context of size 6, containing three words to the left and three words to the right. We also used the POS tags of words in the context window as features. In future, we plan to use the words syntactically dependent on a source word as global context(shown in Figure 1).

3.1.2 Context Dependent Block Distortion Model

An IBM model 3 like distortion model is trained to predict the relative position of a source word in the target given its context. Given a source word and its context, the model estimates the probability of particular relative position being an appropriate position of the source word in the target (see Figure 2).

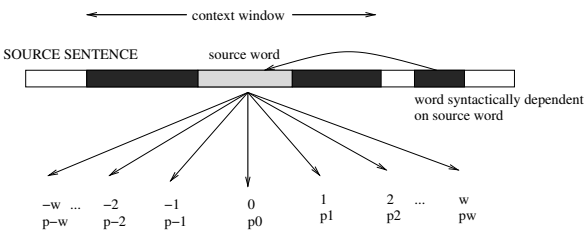


Figure 2: Position prediction model

Using a maximum entropy model similar to

the one described in the context dependent block translation model, binary classifiers are trained for every possible relative position in the target. These classifiers output a probability distribution over various relative positions given a source word and its context.

The word alignments in the training corpus are used to train the distortion model. While computing the relative position, the difference in sentence lengths is also taken into account. Hence, the relative position of the target block located at position i corresponding to the source word located at position j is given in equation 6.

$$r = \textbf{round}(i * \frac{m}{n} - j) \quad (6)$$

where, m is the length of source sentence and n is the number of target blocks. **round** is the function to compute the nearest integer of the argument. If the source word is not aligned to any target word, a special symbol 'INF' is used to indicate such a case. In our model, this symbol is also a part of the target distribution.

The features used to train this model are the same as those used for the block translation model. In order to use further lexical information, we also incorporated information about the target word for predicting the distribution. The information about possible target words is obtained from the 'context dependent block translation model'. The probabilities in this case are measured as shown in equation 7

$$p_{r,e_i} = P(true|r, e_i, f_j, C) \quad (7)$$

3.2 Integration with phrase-based decoder

The discriminative models trained are sentence specific, i.e. the context of the sentence is used to make predictions in these models. Hence, the phrase-based decoder is required to use information specific to a source sentence. In order to handle this issue, a different phrase-table and reordering-table are constructed for every input sentence. The phrase-table and reordering-table are constructed using the discriminative models trained earlier.

In Moses (Koehn et al., 2007), the phrase-table contains the source phrase, the target phrase and the various scores associated with the phrase pair such as phrase translation probability, lexical weighting, inverse phrase translation probability, etc.[1]

In our approach, given a source sentence, the following steps are followed to construct the phrase table.

1. Extract source blocks ('words' in this work)
2. Use the 'context dependent block translation model' to predict the possible target blocks. The set of possible blocks can be predicted using two criteria, (1) Probability threshold, and (2) K-best. Here, we use a threshold value to prune the set of possible candidates in the target vocabulary.
3. Use the prediction probabilities to assign scores to the phrase pairs.

A similar set of steps is used to construct the reordering-table corresponding to an input sentence in the source language.

4 Decoding

4.1 Decoding with LCS Decoder

The lattice construction and scoring algorithm, as the name suggests, consists of two steps,

1. Lattice construction

   In this step, a lattice representing various possible target sequences is obtained. In the approach for global lexical selection (Bangalore et al., 2007; Venkatapathy and Bangalore, 2009), the input to this step is a bag of words. The bag of words is used to construct an initial sequence (a single path lattice). To this sequence, deletion arcs are added to incorporate additional paths (at a cost) that facilitate deletion of words in the initial sequence. This sequence is permuted using a permutation window in order to construct a lattice representing possible sequences. The permutation window is used to control the search space.

   In our experiments, we used a similar process for sentence construction. Using the context dependent block translation algorithm,

[1] http://www.statmt.org/moses/?n=FactoredTraining.ScorePhrases

we obtain a number of translation blocks for every source word. These blocks are interconnected in order to obtain the initial lattice (see figure 3).

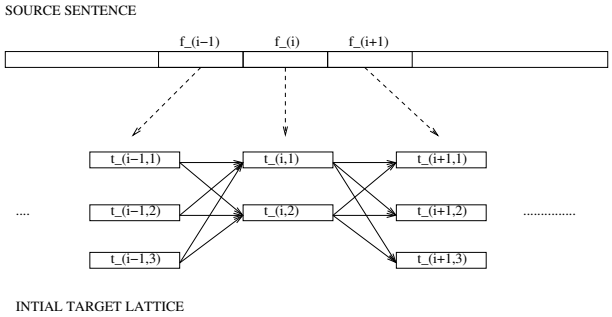


Figure 3: Lattice Construction

To control deletions at various source positions, deletion nodes may be added to the initial lattice. This lattice is permuted using a permutation window to construct a lattice representing possible sequences. Hence, the parameters that dictate lattice construction are, (1) Threshold for lexical selection, (2) Using deletion arcs or not, and (3) Permutation window.

2. Scoring

   In this step, each of the paths in the lattice constructed in the earlier step is scored using a language model (Haffner, 2006), which is same as the one used in the sentence construction in global lexical selection models. It is to be noted that we do not use the discriminative reordering model in this decoder, and only the language model is used to score various target sequences.

The path with the lowest score is considered the best possible target sentence for the given source sentence. Using this decoder, we conducted experiments on the development set by varying threshold values and the size of the permutation window. The best parameter values obtained using the development set were used for decoding the test corpus.

4.2 Decoding with Moses Decoder

In this approach, the phrase-table and the reordering-table are constructed using the discriminative model for every source sentence (see section 3.2). These tables are then used by the state-of-art Moses decoder to obtain corresponding translations.

The various training and decoding parameters of the discriminative model are computed by exhaustively exploring the parameter space, and correspondingly measuring the output quality on the development set. The best set of parameters were used for decoding the sentences in the test corpus. We modified the weights assigned by MOSES to the translation model, reordering model and language model. Experiments were conducted by performing pruning on the options in the phrase table and by using the word penalty feature in MOSES.

We trained a language model of order 5 built on the entire EUROPARL corpus using the SRILM package. The method uses improved Kneser-Ney smoothing algorithm (Chen and Goodman, 1999) to compute sequence probabilities.

5 Dataset

The experiments were conducted on the Spanish-English language pair. The latest version of the Europarl corpus(version-5) was used in this work. A small set of 200K sentences was selected from the training set to conduct the experiments. The test and development sets containing 2525 sentences and 2051 sentences respectively were used, without making any changes.

Corpus	No. of sentences	Source	Target
Training	200000	59591	36886
Testing	2525	10629	8905
Development	2051	8888	7750
Monolingual English (LM)	200000	n.a	36886

Table 1: Corpus statistics for Spanish-English corpus.

6 Experiments and Results

The output of our experiments was evaluated using two metrics, (1) BLEU (Papineni et al., 2002), and (2) Lexical Accuracy (LexAcc). Lexical accuracy measures the similarity between the unordered bag of words in the reference sentence

against the unordered bag of words in the hypothesized translation. Lexical accuracy is a measure of the fidelity of lexical transfer from the source to the target sentence, independent of the syntax of the target language (Venkatapathy and Bangalore, 2009). We report lexical accuracies to show the performance of LCS decoding in comparison with the baseline system.

We first present the results of the state-of-art phrase-based model (Moses) trained on a parallel corpus. We treat this as our baseline. The reordering feature used is msd-bidirectional, which allows for all possible reorderings over a specified distortion limit. The baseline accuracies are shown in table 2.

Corpus	BLEU	Lexical Accuracy
Development	0.1734	0.448
Testing	0.1823	0.492

Table 2: Baseline Accuracy

We conduct two types of experiments to test our approach.

1. Experiments using lexical features (see section 6.1), and
2. Experiments using syntactic features (see section 6.2).

6.1 Experiments using Lexical Features

In this section, we present results of our experiments that use only lexical features. First, we measure the translation accuracy using LCS decoding. On the development set, we explored the set of decoding parameters (as described in section 4.1) to compute the optimal parameter values. The best lexical accuracy obtained on the development set is **0.4321** and the best BLEU score obtained is **0.0923** at a threshold of 0.17 and a permutation window size of value 3. The accuracies corresponding to a few other parameter values are shown in Table 3.

On the test data, we obtained a lexical accuracy of **0.4721** and a BLEU score of **0.1023**. As we can observe, the BLEU score obtained using the LCS decoding technique is low when compared to the BLEU score of the state-of-art system. However, the lexical accuracy is comparable to the lexical accuracy of Moses. This shows that the discriminative model provides good lexical selection, while the sentence construction technique does not perform as expected.

Threshold	Perm. Window	LexAcc	BLEU
0.16	3	0.4274	0.0914
0.17	3	0.4321	0.0923
0.18	3	0.4317	0.0918
0.16	4	0.4297	0.0912
0.17	4	0.4315	0.0915

Table 3: Lexical Accuracies of Lattice-Output using lexical features alone for various parameter values

Next, we present the results of the Moses based decoder that uses the discriminative model (see section 3.2). In our experiments, we did not use MERT training for tuning the Moses parameters. Rather, we explore a set of possible parameter values (i.e. weights of the translation model, reordering model and the language model) to check the performance. We show the BLEU scores obtained on the development set using Moses decoder in Table 4.

Reordering weight(d)	LM weight(l)	Translation weight(t)	BLEU
0	0.6	0.3	0.1347
0	0.6	0.6	0.1354
0.3	0.6	0.3	0.1441
0.3	0.6	0.6	0.1468

Table 4: BLEU for different weight values using lexical features only

On the test set, we obtained a BLEU score of **0.1771**. We observe that both the lexical accuracy and the BLEU scores obtained using the discriminative training model combined with the Moses decoder are comparable to the state-of-art results. The summary of the results obtained using three approaches and lexical feature functions is presented in Table 5.

6.2 Experiments using Syntactic Features

In this section, we present the effect of incorporating syntactic features using our model on the

Approach	BLEU	LexAcc
State-of-art(MOSES)	0.1823	0.492
LCS decoding	0.1023	0.4721
Moses decoder trained using a discriminative model	0.1771	0.4841

Table 5: Translation accuracies using lexical features for different approaches

translation accuracies. Table 6 presents the results of our approach that uses syntactic features at different parameter values. Here, we can observe that the translation accuracies (both LexAcc and BLEU) are better than the model that uses only lexical features.

Reordering weight(d)	LM weight(l)	Translation weight(t)	BLEU
0	0.6	0.3	0.1661
0	0.6	0.6	0.1724
0.3	0.6	0.3	0.1780
0.3	0.6	0.6	0.1847

Table 6: BLEU for different weight values using syntactic features

Table 7 shows the comparative performance of the model using syntactic as well as lexical features against the one with lexical features functions only.

Model	BLEU	LexAcc
Lexical features	0.1771	0.4841
Lexical+Syntactic features	**0.201**	**0.5431**

Table 7: Comparison between translation accuracies from models using syntactic and lexical features

On the test set, we obtained a BLEU score of **0.20** which is an improvement of **2.3** points over the model that uses lexical features alone. We also obtained an increase of **6.1%** in lexical accuracy using this model with syntactic features as compared to the model using lexical features only.

7 Conclusions and Future Work

In this paper, we presented an approach to statistical machine translation that combines the power of a discriminative model (for training a model for Machine Translation), and the standard beam-search based decoding technique (for the translation of an input sentence). The key contributions are:

1. We incorporated a discriminative model in a phrase-based decoder. We obtained comparable results with the state-of-art phrase-based decoder (see section 6.1). The advantage in using our approach is that it has the flexibility to incorporate richer contextual and linguistic feature functions.

2. We show that the incorporation of syntactic information (POS tags) in our discriminative model boosted the performance of translation. The lexical accuracy using our approach improved by **6.1%** when syntactic features were used in addition to the lexical features. Similarly, the BLEU score improved by **2.3** points when syntactic features were used compared to the model that uses lexical features alone. The accuracies are likely to improve when richer linguistic feature functions (that use parse structure) are incorporated in our approach.

In future, we plan to work on:

1. Experiment with rich syntactic and structural features (parse tree-based features) using our approach.

2. Experiment on other language pairs such as Arabic-English and Hindi-English.

3. Improving LCS decoding algorithm using syntactic cues in the target (Venkatapathy and Bangalore, 2007) such as supertags.

References

Bangalore, S., P. Haffner, and S. Kanthak. 2007. Statistical machine translation through global lexical selection and sentence reconstruction. In *Annual Meeting-Association for Computational Linguistics*, volume 45, page 152.

Berger, A.L., V.J.D. Pietra, and S.A.D. Pietra. 1996. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Brown, P.F., V.J.D. Pietra, S.A.D. Pietra, and R.L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Chen, S.F. and J. Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–394.

Haffner, P. 2006. Scaling large margin classifiers for spoken language understanding. *Speech Communication*, 48(3-4):239–261.

Hassan, H., K. Sima'an, and A. Way. 2009. A syntactified direct translation model with linear-time decoding. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1182–1191. Association for Computational Linguistics.

Ittycheriah, A. and S. Roukos. 2007. Direct translation model 2. In *Proceedings of NAACL HLT*, pages 57–64.

Koehn, P. and H. Hoang. 2007. Factored translation models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876.

Koehn, P., F.J. Och, and D. 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. Association for Computational Linguistics.

Koehn, P., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual meeting-association for computational linguistics*, volume 45, page 2.

Och, F.J. and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL*, volume 2, pages 295–302.

Och, F.J., C. Tillmann, H. Ney, et al. 1999. Improved alignment models for statistical machine translation. In *Proc. of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 20–28.

Papineni, KA, S. Roukos, and RT Ward. 1998. Maximum likelihood and discriminative training of directtranslation models. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, volume 1.

Papineni, K., S. Roukos, T. Ward, and W.J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Venkatapathy, S. and S. Bangalore. 2007. Three models for discriminative machine translation using Global Lexical Selection and Sentence Reconstruction. In *Proceedings of the NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation*, pages 96–102. Association for Computational Linguistics.

Venkatapathy, Sriram and Srinivas Bangalore. 2009. Discriminative Machine Translation Using Global Lexical Selection. *ACM Transactions on Asian Language Information Processing*, 8(2).

Xiong, D., Q. Liu, and S. Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, page 528. Association for Computational Linguistics.

Seeding Statistical Machine Translation with Translation Memory Output through Tree-Based Structural Alignment

Ventsislav Zhechev
EuroMatrixPlus, CNGL
School of Computing, Dublin City University
contact@VentsislavZhechev.eu

Josef van Genabith
EuroMatrixPlus, CNGL
School of Computing, Dublin City University
josef@computing.dcu.ie

Abstract

With the steadily increasing demand for high-quality translation, the localisation industry is constantly searching for technologies that would increase translator throughput, with the current focus on the use of high-quality Statistical Machine Translation (SMT) as a supplement to the established Translation Memory (TM) technology. In this paper we present a novel modular approach that utilises state-of-the-art sub-tree alignment to pick out pre-translated segments from a TM match and seed with them an SMT system to produce a final translation. We show that the presented system can outperform pure SMT when a good TM match is found. It can also be used in a Computer-Aided Translation (CAT) environment to present almost perfect translations to the human user with markup highlighting the segments of the translation that need to be checked manually for correctness.

1. Introduction

As the world becomes increasingly interconnected, the major trend is to try to deliver ideas and products to the widest audience possible. This requires the localisation of products for as many countries and cultures as possible, with translation being one of the main parts of the localisation process. Because of this, the amount of data that needs professional high-quality translation is continuing to increase well beyond the capacity of the world's human translators.

Thus, current efforts in the localisation industry are mostly directed at the reduction of the amount of data that needs to be translated from scratch by hand. Such efforts mainly include the use of Translation Memory (TM) systems, where earlier translations are stored in a database and offered as suggestions when new data needs to be translated. As TM systems were originally limited to providing translations only for (almost) exact matches of the new data, the integration of Machine Translation (MT) techniques is seen as the only feasible development that has the potential to significantly reduce the amount of manual translation required.

At the same time, the use of SMT is frowned upon by the users of CAT tools as they still do not trust the quality of the SMT output. There are two main reasons for that. First, currently there is no reliable way to automatically ascertain the quality of SMT-generated translations, so that the user could at a glance make a judgement as to the amount of effort that might be needed to post-edit the suggested translation (Simard and Isabelle, 2009). Not having such automatic quality metrics also has the side effect of it being impossible for a Translation-Services Provider (TSP) company to reliably determine in advance the increase in translator productivity due to the use of MT and to adjust their resources-allocation and cost models correspondingly.

The second major problem for users is that SMT-generated translations are as a rule only obtained for cases where the TM system could not produce a good-enough translation (cf. Heyn, 1996). Given that the SMT system used is usually trained only on the data available in the TM, expectedly it also has few examples from which to construct the translation, thus producing low quality output.

In this paper, we combine a TM, SMT and an automatic Sub-Tree Alignment (STA) backends in a single integrated tool. When a new sentence that needs to be translated is supplied, first a Fuzzy-Match Score (FMS – see Section 2.2) is obtained from the TM backend, together with the suggested matching sentence and its translation. For sentences that receive a reasonably high FMS, the STA backend is used to find the correspondences between the input sentence and the TM-suggested translation, marking up the parts of the input that are correctly translated by the TM. The SMT backend is then employed to obtain the final translation from the marked-up input sentence. In this way we expect to achieve a better result compared to using pure SMT.

In Section 2, we present the technical details of the design of our system, together with motivation for the particular design choices. Section 3 details the experimental setup and the data set used for the evaluation results in Section 4. We present improvements that we plan to investigate in further work in Section 5, and provide concluding remarks in Section 6.

2. System Framework

We present a system that uses a TM-match to pre-translate parts of the input sentence and guide an SMT system to the generation of a higher-quality translation.

2.1. Related Approaches

We are not aware of any published research where TM output is used to improve the performance of an SMT system in a manner similar to the system presented in this paper.

Most closely related to our approach are the systems by Biçici and Dymetman (2008) and Simard and Isabelle (2009), where the authors use the TM output to extract new phrase pairs that supplement the SMT phrase table. Such an approach, however, does not guarantee that the SMT system will select the TM-motivated phrases even if a heavy bias is applied to them.

Another related system is presented in (Smith and Clark, 2009). Here the authors use a syntax-based EBMT system to pre-translate and mark-up parts of the input sentence and then supply this marked-up input to an SMT system. This differs to our system in two ways. First, Smith and Clark use EMBT techniques to obtain partial translations of the input from the complete example base, while we are only looking at the best TM match for the given input. Second, the authors use dependency structures for EMBT matching, while we employ phrase-based structures.

2.2. Translation Memory Backend

Although the intention is to use a full-scale TM system as the translation memory backend, to have complete control over the process for this initial research we decided to build a simple prototype TM backend ourselves.

We employ a database setup using the PostgreSQL v.8.4.3[1] relational database management (RDBM) system. The segment pairs from a given TM are stored in this database and assigned unique IDs for further reference. When a new sentence is supplied for translation, the database is searched for (near) matches, using an FMS based on normalised character-level Levenshtein edit distance (Levenshtein, 1965).

Thus for each input sentence, from the database we obtain the matching segment with the highest FMS, its translation and the score itself.

2.3. Sub-Tree Alignment Backend

The system presented in this paper uses phrase-based sub-tree structural alignment (Zhechev, 2010) to discover parts of the input sentence that correspond to parts of the suggested translation extracted from the TM database. We chose this particular tool, because it can produce aligned phrase-based-tree pairs from unannotated (i.e. unparsed) data. It can also function fully automatically without the need for any training data. The only auxiliary requirement it has is for a probabilistic dictionary for the languages that are being aligned. As described later in this section, in our case this is obtained automatically from the TM data during the training of the SMT backend.

The matching between the input sentence and the TM-suggested translation is done in a three-step process. First, the plain TM match and its

[1] http://www.postgresql.org/

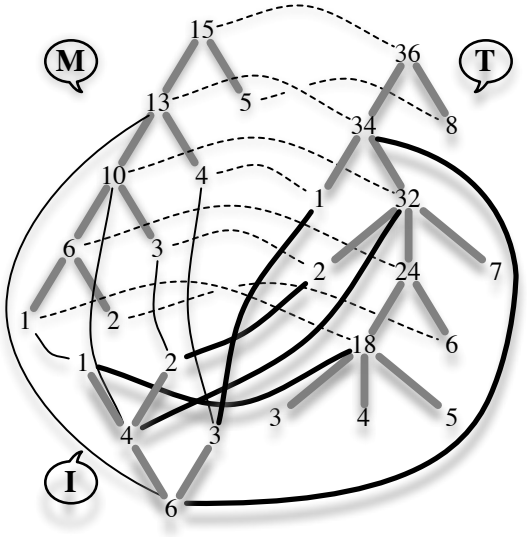
translation are aligned, which produces a sub-tree-aligned phrase-based tree pair with all non-terminal nodes labelled '*X*' (cf. Zhechev, 2010). As we are only interested in the relations between the lexical spans of the non-terminal nodes, we can safely ignore their labels. We call this first step of our algorithm *bilingual alignment*.

In the second step, called *monolingual alignment*, the phrase-based tree-annotated version of the TM match is aligned to the unannotated input sentence. The reuse of the tree structure for the TM match allows us to use it in the third step as an intermediary to establish the available sub-tree alignments between the input sentence and the translation suggested from the TM.

During this final alignment, we identify matched and mismatched portions of the input sentence and their possible translations in the TM suggestion and, thus, this step is called *matching*. Additionally, the sub-tree alignments implicitly provide us with reordering information, telling us where the portions of the input sentence that we translate should be positioned in the final translation.

The alignment process is exemplified in Figure 1. The tree marked 'I' corresponds to the input sentence, the one marked 'M' to the TM match and the one marked 'T' to the TM translation. Due to space constraints, we only display the node ID numbers of the non-terminal nodes in the phrase-structure trees — in reality all nodes carry the label '*X*'. These IDs are used to identify the sub-sentential alignment links. The lexical items corresponding to the leaves of the trees are presented in the table below the graph.

The alignment process can be visually represented as starting at a linked node in the **I** tree and following the link to the **M** tree. Then, if available, we follow the link to the **T** tree and this leads us to the **T**-tree node corresponding to the **I**-tree node we started from. In Figure 1, this results in the **I**–**T** alignments *I1–T18*, *I2–T2*, *I3–T1*, *I4–T32* and *I6–T34*. The first three links are matches, because the lexical items covered by the **I** nodes correspond exactly to the lexical items covered by their **M** node counterparts. Such alignments provide us with direct TM translations for our input. The last two links in the group are mismatched, because there is no lexical correspondence between the **I** and **M** nodes (node *I4* corresponds to the phrase *sender email*, while the linked node *M10* corresponds to *sender 's email*). Such alignments can only be used to infer reordering information. In particular in this case, we can infer that the target word order for the input sentence is *address email sender*, which produces the translation *adresse électronique de l' expéditeur*.



I	1	2	3					
input	sender	email	address					
M	1	2	3	4	5			
match	sender	's	email	address	.			
T	1	2	3	4	5	6	7	8
trans-lation	adresse	électro-nique	de	l'	expé-diteur	du	mes-sage	.

Figure 1. Example of sub-tree alignment between an input sentence, TM match and TM translation

We decided to use sub-tree-based alignment, rather than plain word alignment (e.g. GIZA++ – Och and Ney, 2003), due to a number of factors. First, sub-tree-based alignment provides much better handling of long-distance reorderings, while word– and phrase-based alignment models always have a fixed limit on reordering distance that tends to be relatively low to allow efficient computation.

The alignments produced by a sub-tree alignment model are also precision-oriented, rather than recall-oriented (cf. Tinsley, 2010). This is important in our case, where we want to only extract those parts of the translation suggested by the TM for which we are most certain that they are good translations.

As stated earlier, the only resource necessary for the operation of this system is a probabilistic bilingual dictionary covering the data that needs to be aligned. For the *bilingual alignment* step, such a bilingual dictionary is produced as a by-product of the training of the SMT backend and therefore available. For the *monolingual alignment* step, the required probabilistic dictionary is generated by simply listing each unique token seen in the source-language data in the TM as translating only as itself with probability 1.

2.4. Statistical Machine Translation Backend

Once the *matching* step is completed, we have identified and marked-up the parts of the input sentence for which translations will be extracted from the TM suggestions, as well as the parts that need to be translated from scratch. The lengths of the non-translated segments vary depending on the FMS, but are in general relatively short (one to three tokens).

The further processing of the input relies on a specific feature of the SMT backend we use, namely the Moses system (Koehn et al., 2007). We decided to use this particular system as it is the most widely adopted open-source SMT system, both for academic and commercial purposes. In this approach, we annotate the segments of the input sentence for which translations have been found from the TM suggestion using XML tags with the translation corresponding to each segment given as an attribute to the encapsulating XML tag, similarly to the system described in (Smith and Clark, 2009). The SMT backend is supplied with marked-up input in the form of a string consisting of the concatenation of the XML-enclosed translated segments and the plain non-translated segments in the target-language word order, as established by the alignment process. The SMT backend is instructed to translate this input, while keeping the translations supplied via the XML annotation. This allows the SMT backend to produce translations informed by and conforming to actual examples from the TM, which should result in improvements in translation quality.

2.5. Auxilliary Tools

It must be noted that in general the SMT backend sees the data it needs to translate in the target-language word order (e.g. it is asked to translate an English sentence that has French word order). This, however, does not correspond to the data found in the TM, which we use for deriving the SMT models. Because of this discrepancy, we developed a pre-processing tool that goes over the TM data performing *bilingual alignment* and outputting reordered versions of the sentences it processes by using the information implicitly encoded in the sub-tree alignments. In this way we obtain the necessary reordered data to train a translation model where the source language already has the target-language word order. In our system we than use this model — together with the proper-word-order model — for translation.

One specific aspect of real-world TM data that we need to deal with is that they often contain meta-tag annotations of various sorts. Namely, annotation tags specific to the file format used for storing the TM data, XML tags annotating parts of the text as appearing in Graphical User Interface (GUI) elements, formatting tags specific to the file format the TM data was originally taken from, e.g. RTF, OpenDoc, etc. Letting any MT system try to deal with these tags in a probabilistic manner can easily result in ill-formed, mistranslated and/or out-of-order meta-tags in the translation.

This motivates the implementation of a rudimentary handling of meta-tags in the system presented in this paper, in particular handling the XML tags found in the TM data we work with, as described in Section 3. The tool we developed for this purpose simply builds a map of all unique XML tags per language and replaces them in the data with short placeholders that are designed in such a way that they would not interfere with the rest of the TM data.[2] A special case that the tool has to take care of is when an XML tag contains an attribute whose value needs to be translated. In such situations, we decided to not perform any processing, but rather leave the XML tag as is, so that all text may be translated as needed. A complete treatment of meta-tags, however, is beyond the scope of the current paper.

[2] In the current implementation, the XML tags are replaced with the string <tag_id>, where <tag_id> is a unique numeric identifier for the XML tag that is being replaced.

We also had to build a dedicated tokeniser/detokeniser pair to handle real world TM data containing meta-tags, e-mail addresses, file paths, etc., as described in Section 3. Both tools are implemented as a cascade of regular expression substitutions in Perl.

Finally, we use a tool to extract the textual data from the TM. That is, we strip all tags specific to the format in which the TM is stored, as they can in general be recreated and thus do not need to be present during translation. In our particular case the TM is stored in the XML-based TMX format.[3]

2.6. Complete Workflow

Besides the components described above, we also performed two further transformations on the data. First, we lowercase the TM data before using it to train the SMT backend models. This also means that the alignment steps from Section 2.3 are performed on lowercased data, as the bilingual dictionary used there is obtained during the SMT training process.[4]

Additionally, the SMT and sub-tree alignment systems that we use cannot handle certain characters, which we need to mask in the data. For the SMT backend, this includes '|', '<' and '>' and for the sub-tree aligner, '(' and ')'. The reason these characters cannot be handled is that the SMT system uses '|' internally to separate data fields in the trained models and '<' and '>' cannot be handled whilst using XML tags to annotate pre-translated portions of the input. The sub-tree aligner uses '(' and ')' to represent the phrase-based tree structures it generates and the presence of these characters in the data may create ambiguity when parsing the tree structures. All these characters are masked by substituting in high-Unicode counterparts, namely '|', '<', '>', '(' and ')'. Visually, there is a very slight distinction and this is intentionally so to simplify debugging. However, the fact that the character codes are different alleviates the problems discussed above. Of course, in the final output, the masking is reversed and the translation contains the regular versions of the characters.

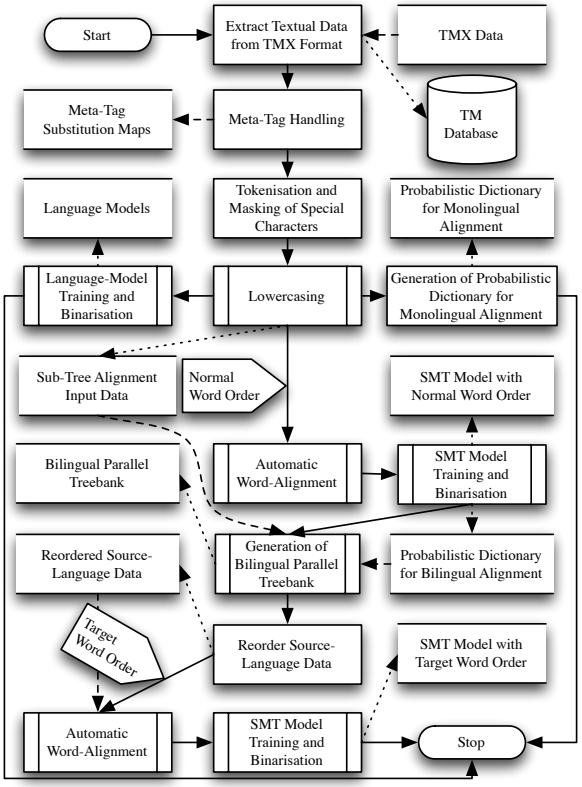


Figure 2. Pre-Processing Workflow

The complete pre-processing workflow is presented in Figure 2, where the rectangles with vertical bars represent the use of open-source tools, while the plain rectangles represent tools developed by the authors of this paper.

First, the textual data is extracted from the original TM format, producing one plain-text file for each language side. These data can either be pre-loaded in a PostgreSQL database at this time, or during the first run of the translation system.

Next, the meta-tag-handling tool is used to generate the substitution tables for the source and target languages, as well as new files for each language with the tags substituted by the corresponding identifiers (cf. Section 2.5). These files are then tokenised, lowercased and all conflicting characters are masked, as described above.

The pre-processed files are then used to produce a file containing pairs of sentences in the input format of the sub-tree aligner, as well as to generate the probabilistic dictionary required for

[3] `http://www.lisa.org/fileadmin/standards/tmx1.4/tmx.htm`

[4] Currently, we do not use a recaser tool and the translations produced are always in lowercase. This component, however, will be added in a future version of the system.

the *monolingual alignment* and to train the SMT model on the data in the proper word order. The SMT training produces the necessary bilingual dictionary for use by the sub-tree aligner, which is run to obtain a parallel-treebank version of the TM data. The parallel treebank is then used to retrieve bilingual alignments for the TM data, rather than generate them on the fly during translation. This is an important design decision, as the complexity of the alignment algorithm is high for plain-text alignment (cf. Zhechev, 2010).

Once we have generated the bilingual parallel treebank, we run the reordering tool, which generates a new plain-text file for the source language, where the sentences are modified to conform to the target-language word order, as implied by the data in the parallel treebank. This is then matched with the proper-order target-language file to train the SMT backend for the actual use in the translation process.

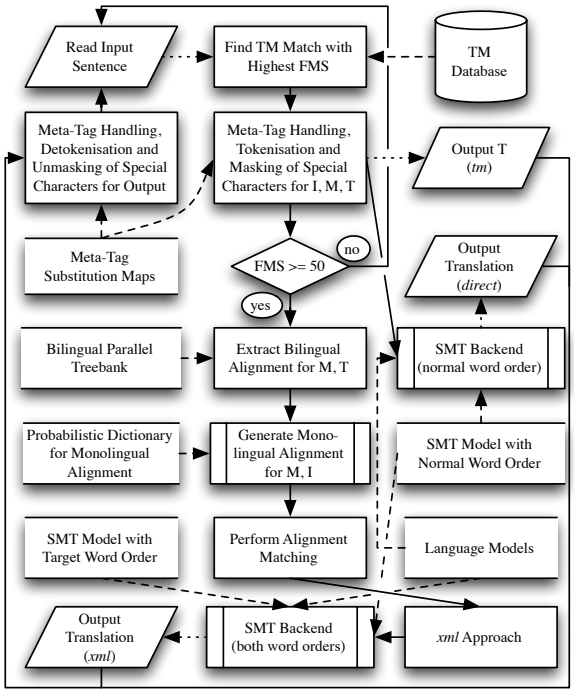


Figure 3. Translation Workflow

Once all the necessary files have been generated and all pre-processing steps have been completed, the system is ready for use for translation. The translation workflow is shown in Figure 3, 'I', 'M' and 'T' having the same meanings as in Figure 1. Here, the first step after an input sentence has been read in is to find the TM match with the highest FMS. This is done using the original plain non-pre-processed data to simulate real-life operation with a proper TM backend.

After the best TM match and its translation are extracted from the TM, they — together with the input sentence — are pre-processed by tokenisation, lowercasing, meta-tag and special-character substitution. Next, the corresponding tree pair is extracted from the bilingual parallel treebank to establish the tree structure for the TM source-language match. This tree structure is then used to perform the *monolingual alignment*, which allows us to perform the *matching* step next. After the *matching* is complete, we generate a final translation as described in Section 2.4. Finally, the translations are de-tokenised and the XML tags and special characters are unmasked.

3. Experimental Setup

We use real-life TM data from an industrial partner. The TM was generated during the translation of RTF-formatted customer support documentation. The data is in TMX format and originally contains 108 967 English–French translation segments, out of which 14 segments either have an empty language side or have an extreme discrepancy in the number of tokens for each language side and were therefore discarded.

A particular real-life trait of the data is the presence of a large number of XML tags. Running the tag-mapping tool described in Section 2.6, we gathered 2 049 distinct tags for the English side of the data and 2 653 for the French side. Still, there were certain XML tags that included a `label` argument whose value was translated from one language to the other. These XML tags were left intact so that our system could handle the translation correctly.

The TM data also contain a large number of file paths, e-mail addresses, URLs and others, which makes bespoke tokenisation of the data necessary. Our tokenisation tool ensures that none of these elements are tokenised, keeps RTF formatting sequences non-tokenised and properly handles non-masked XML tags, minimising their fragmentation.

As translation segments rarely occur more than once in a TM, we observe a high number of unique tokens (measured after pre-processing) — 41 379 for English and 49 971 for French — out of

108 953 segment pairs. The average sentence length is 13.2 for English and 15.0 for French.

For evaluation, we use a data set of 4977 English–French segments from the domain of the TM. The sentences in the test set are significantly shorter on average, compared to the TM — 9.2 tokens for English and 10.9 for French.

It must be noted that we used SMT models with maximum phrase length of 3 tokens, rather than the standard 5 tokens, and for decoding we used a 3-gram language model. This results in much smaller models than the ones usually used in mainstream SMT applications. (The standard for some tools goes as far as 7-token phase-length limit and 7-gram language models)

4. Evaluation Results

For the evaluation of our system, we used a number of widely accepted automatic metrics, namely BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), TER (Snover et al., 2006) and inverse F-Score based on token-level precision and recall.

We setup our system to only fully process input sentences for which a TM match with an FMS over 50% was found, although all sentences were translated directly using the SMT backend to check the overall pure SMT performance. The TM-suggested translations were also output for all input sentences.

The results of the evaluation are given in Figure 4, where the *tm* and *direct* scores are also given for the FMS range [0%; 50%)∪{100%}. Across all metrics we see a uniform drop in the quality of TM-suggested translations, which is what we expected, given that these translations contain one or more wrong words. We believe that the relatively high scores recorded for the TM-suggested translations at the high end of the FMS scale are a result of the otherwise perfect word order and lexical choice. For *n*-gram-match-based metrics like the ones we used such a result is expected and predictable. Although the inverse F-score results show the potential of our setup to translate the outstanding tokens in a 90%–100% TM match, it appears that the SMT system produces word order that does not correspond to the reference translation and because of this receives lower scores on the other metrics.

The unexpected drop in scores for perfect TM matches is due to discrepancies between the reference translations in our test set and the translations stored in the TM. We believe that this issue

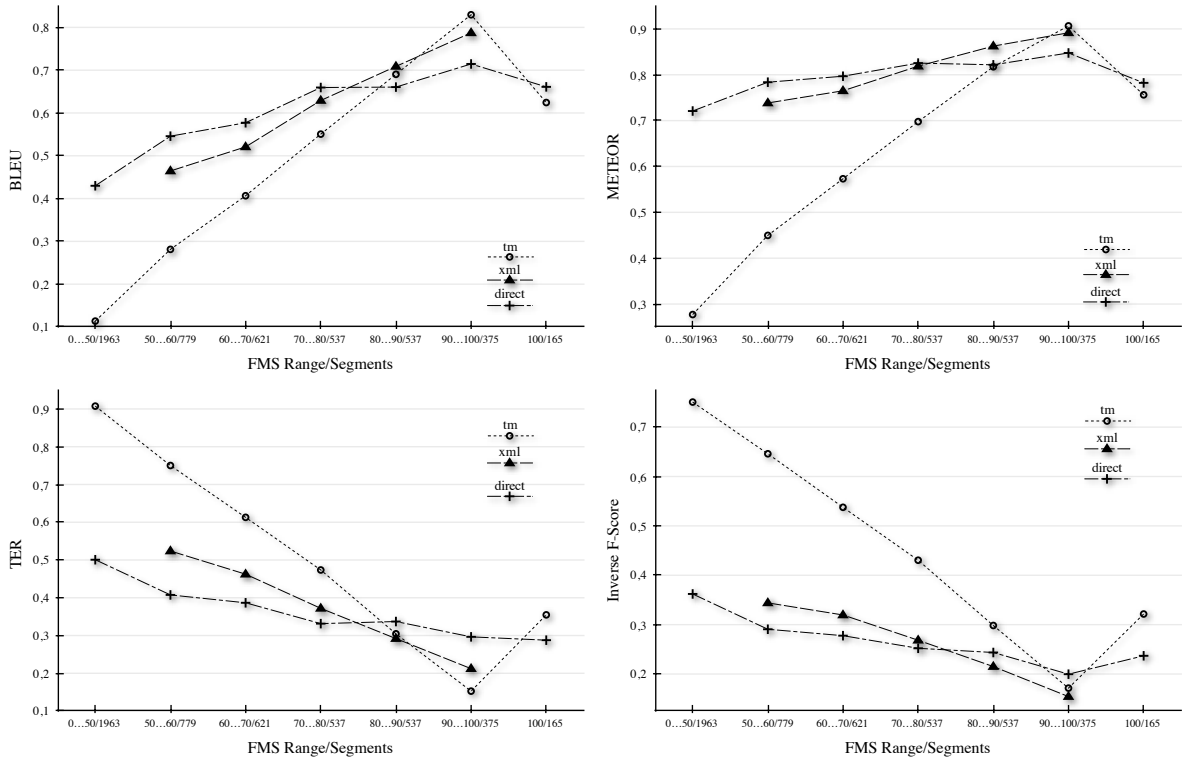


Figure 4. Evaluation results for English-to-French translation, broken down by FMS range

affects all FMS ranges, albeit to a lower extent for non-perfect matches. Unfortunately, the exact impact cannot be ascertained without human evaluation.

We observe a significant drop-off in translation quality for the *direct* output below FMS 50%. This suggests that sentences with such low FMS should be translated either by a human translator from scratch, or by an SMT system trained on different/more data.

Our system (i.e. the *xml* setup) clearly outperforms the *direct* SMT translation for FMS between 80 and 100 and has comparable performance between FMS 70 and 80. Below FMS 70, the SMT backend has the best performance. Although these results are positive, we still need to investigate why our system has poor performance at lower FMS ranges. Theoretically, it should outperform the SMT backend across all ranges, as its output is generated by supplying the SMT backend with good pre-translated fragments. The Inverse F-Score graph suggest that this is due to worse lexical choice, but only manual evaluation can provide us with clues for solving the issue.

The discrepancy in the results in the Inverse F-Score graph with the other metrics suggest that the biggest problem for our system is producing output in the expected word-order.

5. Future Work

There are a number of possible directions for improvement that can be explored.

As mentioned earlier, we plan to integrate our system with a full-featured open-source or commercial TM product that will supply the TM matches and translations. We expect this to improve our results, as the quality of the TM matches will better correspond to the reported FMS.

Such an integration will also be the first necessary step to perform a user study evaluating the effect of the use of our system on post-editing speeds. We expect the findings of such a study to show a significant increase of throughput that will significantly reduce the costs of translation for large-scale projects.

It would be interesting to also conduct a user study where our system is used to additionally mark up the segments that need to be edited in the final SMT translation. We expect this to provide additional speedup to the post-editing process. Such a study will require tight integration between our system and a CAT tool and the modular design we presented will facilitate this significantly.

The proposed treatment of meta-tags is currently very rudimentary and may be extended with additional features and to handle additional types of tags. The design of our system currently allows the meta-tag-handling tool to be developed independently, thus giving the user the choice of using a different meta-tag tool for each type of data they work with.

In addition, the reordering tool needs to be developed further, with emphasis on properly handling situations where the appropriate position of an input-sentence segment cannot be reliably established. In general, further research is needed into the reordering errors introduced by the SMT system into otherwise good translations.

6. Conclusions

In this paper, we presented a novel modular approach to the utilisation of Translation Memory data to improve the quality of Statistical Machine Translation.

The system we developed uses precise subtree-based alignments to reliably determine and mark up correspondences between an input sentence and a TM-suggested translation, which ensures the utilisation of the high-quality translation data stored in the TM database. An SMT backend then translates the marked-up input sentence to produce a final translation with improved quality.

Our evaluation shows that the system presented in this paper significantly improves the quality of SMT output when using TM matches with FMS above 80 and produces results on par with the pure SMT output for SMT between 70 and 80. TM matches with FMS under 70 seem to provide insufficient reordering information and too few matches to improve on the SMT output. Still, further investigation is needed to properly diagnose the drop in quality for FMS below 70.

We expect further improvements to the reordering functionality of our system to result in higher-quality output even for lower FMS ranges.

Acknowledgements

This research is funded under the 7th Framework Programme of the European Commission within the EuroMatrixPlus project (grant № 231720). The data used for evaluation was generously provided by Symantec Ireland.

References

Banerjee, Satanjeev and Alon Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgements. In *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the 43rd Annual Meeting of the Association for Computational Linguistics (ACL '05)*, pp. 65–72. Ann Arbor, MI.

Biçici, Ergun and Marc Dymetman. 2008. Dynamic Translation Memory: Using Statistical Machine Translation to improve Translation Memory Fuzzy Matches. In *Proceedings of the 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing '08)*, ed. Alexander F. Gelbukh, pp. 454–465. Vol. 4919 of *Lecture Notes in Computer Science*. Haifa, Israel: Springer Verlag.

Heyn, Matthias. 1996. Integrating Machine Translation into Translation Memory Systems. In *Proceedings of the EAMT Machine Translation Workshop, TKE '96*, pp. 113–126. Vienna, Austria.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Demo and Poster Sessions of the 45th Annual Meeting of the Association for Computational Linguistics (ACL '07)*, pp. 177–180. Prague, Czech Republic.

Levenshtein, Vladimir I. 1965. Двоичные коды с исправлением выпадений, вставок и замещений символов (Binary Codes Capable of Correcting Deletions, Insertions, and Reversals). *Доклады Академий Наук СССР*, 163 (4): 845–848. [reprinted in: Soviet Physics Doklady, 10: 707–710.].

Och, Franz Josef and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29 (1): 19–51.

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 of Computational Linguistics (ACL '02)*, pp. 311–318. Philadelphia, PA.

Simard, Michel and Pierre Isabelle. 2009. Phrase-based Machine Translation in a Computer-assisted Translation Environment. In *The Twelfth Machine Translation Summit (MT Summit XII)*, pp. 120–127. Ottawa, ON, Canada.

Smith, James and Stephen Clark. 2009. EBMT for SMT: A New EBMT–SMT Hybrid. In *Proceedings of the 3rd International Workshop on Example-Based Machine Translation (EBMT '09)*, eds. Mikel L. Forcada and Andy Way, pp. 3–10. Dublin, Ireland.

Snover, Matthew, Bonnie J. Dorr, Richard Schwartz, Linnea Micciulla and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA '06)*, pp. 223–231. Cambridge, MA.

Tinsley, John. 2010. Resourcing Machine Translation with Parallel Treebanks. School of Computing, Dublin City Univercity: PhD Thesis. Dublin, Ireland.

Zhechev, Ventsislav. 2010. Automatic Generation of Parallel Treebanks. An Efficient Unsupervised System: Lambert Academic Publishing.

Semantic vs. Syntactic vs. N-gram Structure for Machine Translation Evaluation

Chi-kiu LO and **Dekai WU**
HKUST
Human Language Technology Center
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
{jackielo,dekai}@cs.ust.hk

Abstract

We present results of an empirical study on evaluating the utility of the machine translation output, by assessing the accuracy with which human readers are able to complete the semantic role annotation templates. Unlike the widely-used lexical and n-gram based or syntactic based MT evaluation metrics which are fluency-oriented, our results show that using semantic role labels to evaluate the utility of MT output achieve higher correlation with human judgments on adequacy. In this study, human readers were employed to identify the semantic role labels in the translation. For each role, the filler is considered an accurate translation if it expresses the same meaning as that annotated in the gold standard reference translation. Our SRL based f-score evaluation metric has a 0.41 correlation coefficient with the human judgement on adequacy, while in contrast BLEU has only a 0.25 correlation coefficient and the syntactic based MT evaluation metric STM has only 0.32 correlation coefficient with the human judgement on adequacy. Our results strongly indicate that using semantic role labels for MT evaluation can be significantly more effective and better correlated with human judgement on adequacy than BLEU and STM.

1 Introduction

In this paper, we show that evaluating machine translation quality by assessing the accuracy of human performance in reconstructing the semantic frames from the MT output has a higher correlation with human judgment on translation adequacy than (1) the widely-used lexical n-gram precision based MT evaluation metric, BLEU (Papineni *et al.*, 2002), as well as (2) the best-known syntactic tree precision based MT evaluation metric, STM (Liu and Gildea, 2005). At the same time, unlike some highly labor intensive evaluation metrics such as HTER (Snover *et al.*, 2006), our proposed semantic metric only requires simple and minimal instructions to the human judges involved in the evaluation cycle.

We argue that neither n-gram based metrics, like BLEU, nor syntax-based metrics, like STM, adequately capture the similarity in meaning between the machine translation and the reference translation—which, ultimately, is essential for translations to be *useful*.

First, n-gram based metrics assume that "good" translations share the same lexical choices with the reference translation. While BLEU score performs well in capturing the translation *fluency*, Callison-Burch *et al.* (2006) and Koehn and Monz (2006) report cases where BLEU strongly disagrees with human judgment on translation quality. The underlying reason is that lexical similarity does not adequately reflect the similarity in meaning.

Second, just like n-gram based metrics such as BLEU, syntax-based metrics are still more fluency-oriented than adequacy/accuracy-oriented. While STM addresses the failure of BLEU in evaluating the translation *grammaticality*, a grammatical translation can nonetheless achieve a high STM score even if contains errors arising from confusion of semantic roles. Syntactic structure similarity still inadequately reflects similarity of meaning.

As MT systems improve, the shortcomings of

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 52–60,
COLING 2010, Beijing, August 2010.

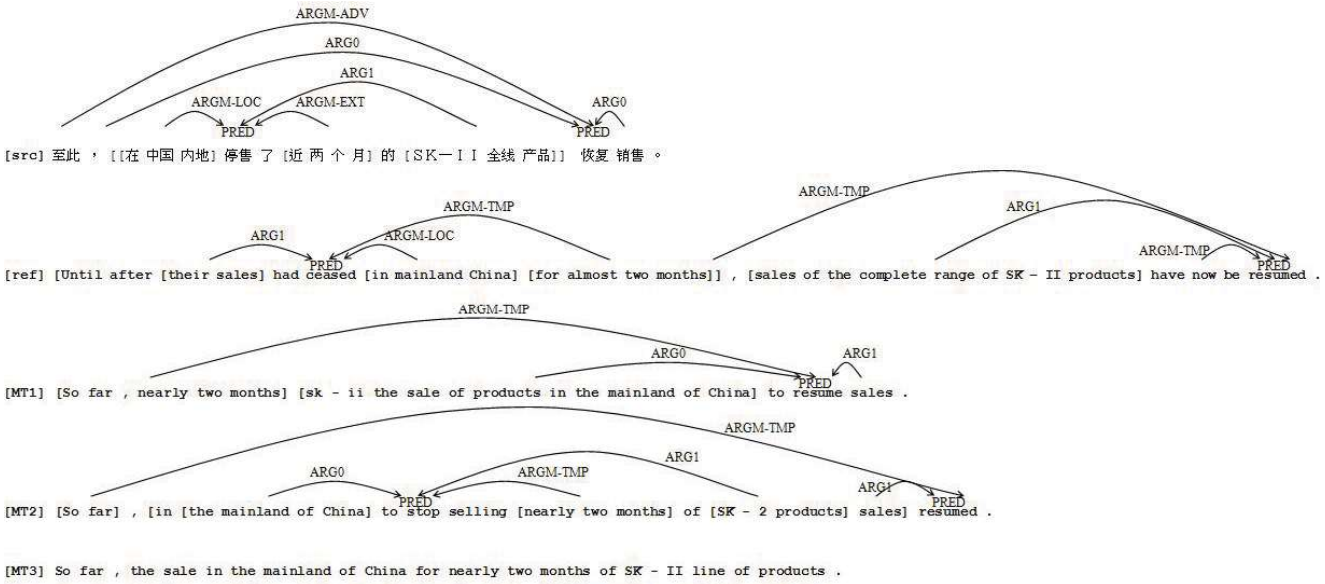


Figure 1: Example of semantic frames in Chinese input, English reference translation and MT output.

lexical n-gram based and syntax-based evaluation metrics are becoming more apparent. State-of-the-art MT systems are often able to output translations containing roughly the correct words and being almost grammatical, but not expressing meaning that is close to the source input. We adopt the outset of the principle that *a good translation is one from which human readers may successfully understand at least the basic event structure* – "who did what to whom, when, where and why" (Pradhan *et al.*, 2004) which represents the most important meaning of the source utterances. Our objective is to evaluate how well the most essential semantic information is being captured by the machine translation systems from the user's point of view.

In this paper, we describe in detail the methodology that underlies the new semantic machine translation evaluation metrics we are developing. We present the results of the study on evaluating machine translation utility by measuring the accuracy with which human readers are able to complete the semantic role annotation templates. Last but not the least, we show that our proposed evaluation metric has a higher correlation with human judgments on adequacy than BLEU and STM.

2 Related Work

2.1 Semantic models in SMT

Numerous recent works has been done on applying different semantic models to statistical machine translation. Word sense disambiguation (WSD) models combine a wide range of context features into a single lexical choice prediction, as in the work of Carpuat and Wu (2007), Chan *et al.* (2007), and Giménez and Màrquez (2007a). In particular, Phrase Sense Disambiguation (PSD), a generalization of the WSD approach, automatically acquires fully phrasal translation lexicons and provides a context-dependent probability distribution over the possible translation candidates for any given phrasal lexicon (Carpuat and Wu, 2007).

Another recent research direction on semantic SMT is applying semantic role labeling models. Semantic role labeling (SRL) is the task of identifying the semantic predicate-argument structures within a sentence. Semantic role labels represent an abstract level of understanding in meaning. There is an increasing availability of large parallel corpora annotated with semantic role information, in particular, in the work of Palmer *et al.* (2005) and Xue and Palmer (2005). As a result, the accuracy of automatic SRL task is also rising.

The best monolingual shallow semantic parser by Fung *et al.* (2006) achieved an F-score of 82.01 in Chinese semantic role labeling, while the best cross-lingual semantic verb frame argument mappings with accuracy of 89.3% as reported in the same work.

The example in Figure 1 is labeled with semantic roles in the Propbank convention. **src** shows a fragment of a typical Chinese source sentence that is drawn from newswire genre of the evaluation corpus. **ref** shows the corresponding fragment of the English reference translation. **MT1**, **MT2** and **MT3** show the three corresponding fragments of the machine translation output from three different MT systems.

A relevant subset of the semantic roles and predicates has been annotated in these fragments, using the PropBank convention of OntoNotes. In the Chinese source sentence, there are two main verbs marked PRED. The first verb "停售" (cease of sales) has three arguments: one in ARG1 experiencer role, "ＳＫ－ＩＩ 全线 产品" (the complete range of SK-II products); one in ARGM-LOC location role, "在 中国 内地" (in mainland China), and one in ARGM-EXT extent role, "近 两 个 月" (for almost two months). The second verb "恢复" (resumed) also has three arguments: two in ARG0 agent roles, "在 中国 内地 停售 了 近 两 个 月 的 ＳＫ－ＩＩ 全线 产品 " (the complete range of SK-II products which sales had ceased in mainland China for almost two months) and "销售" (sales), and one in ARGM-ADV role, "至此" (until then).

In the corresponding English target, there are also two main verbs marked PRED. The first verb (ceased) has three arguments: one in an ARG1 experiencer role, "their sales"; one in an ARGM-LOC role, "in mainland China", and one in ARGM-TMP temporal role, "for almost two months". The second verb (resumed) also has three arguments: two in ARGM-TMP temporal roles, "until after their sales ceased in mainland China for almost two months" and "now", and one in ARG1 experiencer role, "sales of the complete range of SK-II products".

Similarly, the first two MT outputs are also annotated with semantic roles in the PropBank convention. Since there is no verb appeared in the third MT output, no predicate-argument structure is annotated.

Recent work by Wu and Fung (2009a) and Wu and Fung (2009b) has begun to apply SRL to statistical machine translation using a semantic reordering model based on SRL that successfully returns a better translation with fewer semantic role confusion errors.

With recent rise of work applying semantic model to statistical machine translation, there is a high demand for MT evaluation metrics that are directly sensitive to the semantic improvement made. We believe evaluating machine translation utility based on semantic roles should reflect semantic improvement better than current widely-used automated n-gram precision based MT evaluation metrics, like BLEU or fluency-oriented syntactic MT evaluation metrics, like STM.

2.2 STM: syntax-based MT evaluation

Liu and Gildea (2005) proposed to use syntactic features in MT evaluation and developed subtree metric (STM) which based on the similarity of syntax tree of the MT output and that of the reference. It is the first proposed metric that incorporates syntactic features in MT evaluation and underlies all the other recently proposed syntactic MT evaluation metrics.

STM is a precision based metric that captures the fractions of the subtree in a specific depth of the MT output syntax tree which also appear in the reference syntax tree. The fractions of different depths are then average in arithmetic mean.

$$\mathrm{STM} = \frac{1}{D}\sum_{n=1}^{D}\frac{\sum_{t\in \mathrm{subtree}_n(\mathrm{hyp})}\mathrm{count}_{\mathrm{found}}(\mathrm{t})}{\sum_{t\in \mathrm{subtree}_n(\mathrm{hyp})}\mathrm{count}\,(t)}$$

where D is the maximum depth of subtress considered, $\mathrm{count}\,(t)$ denotes the number of times subtree t appears in the MT output's syntax tree, and $\mathrm{count}_{\mathrm{found}}\,(t)$ denotes the found number of times t appears in the references' syntax tree, each subtree in reference will only be found once.

Figure 2 shows the syntax tree of a reference translation and that of the corresponding MT output. For example, we set the maximum depth of subtree considered to 4. There are seven 1-depth subtrees in the MT output (S, NP, VP, PRP, V,

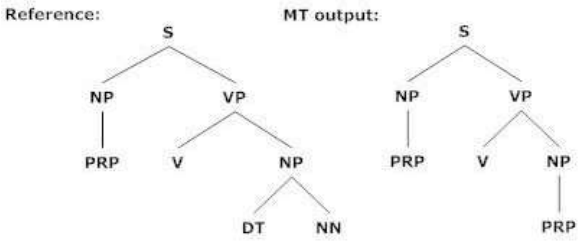


Figure 2: Example for the computation of STM

NP and PRP) in which only six of them appear in the references (S, NP, VP, PRP, V and NP). Note that the found count of PRP should be 1 rather than 2 because there is only one PRP in the reference translation syntax tree. For 2-depth, there are four subtrees in the MT output (S→NP VP, NP→PRP, VP→V NP and NP→PRP) in which three of them appear in the reference (S→NP VP, NP→PRP and VP→V NP). Similarly, there are one out of two 3-depth subtrees and zero out of one 4-depth subtrees in the MT output found in the reference. Therefore, the final STM score for this example is (6/7+3/4+1/2+0/1)/4=0.527.

2.3 MT evaluation metric based on semantic role overlap

Giménez and Màrquez (2008) introduced ULC, a new automatic MT evaluation metric in which a series of linguistic features are combined together. One of those linguistic features is shallow semantic similarity on semantic role overlap. The semantic role overlap metric calculates the lexical overlapping between semantic roles of the same type in the machine translation output and the corresponding reference translations and then considers the average lexical overlapping over all semantic role types.

Despite the fact that the metric shows an improved correlation with human judgment of translation quality (Giménez and Màrquez, 2007b, 2008; Callison-Burch *et al.*, 2007, 2008), it is not commonly used in large-scale MT evaluation campaign. The reason may lie in the high time cost.

We believe it is important to first focus on developing simple measures to evaluate machine translation utility, that make use of *human* extraction of role information. It is necessary to first understand the upper bounds of human performance on this task, as a foundation for better design of efficient automated metrics.

2.4 HTER: non-automated MT evaluation metric

Human-targeted Translation Edit Rate (HTER) in the work of Snover *et al.* (2006) is a non-automatic machine translation evaluation metric based on the number of edits required to correct the translation hypotheses. A human annotator edits each MT hypothesis so that it is meaning-equivalent with the reference translation. It emphasizes on making the minimum possible number of edits. The Translation Edit Rate (TER) is then calculated using the human-edited translation as a targeted reference for the MT hypothesis.

The HTER is highly labor intensive in the evaluation process. The human annotators are not only required to understand the meaning expressed in the reference translation and the machine translation, but are also required to propose minimum possible number of edits to the translation hypotheses. With such heavy-duty human decision requirements, the cost in evaluation is enormously increased, bottlenecking the evaluation cycle. Instead, we believe that any human decisions in the evaluation cycle should be reduced to be as simple as possible.

3 Semantic role translation accuracy

To evaluate the semantic utility of machine translation output, we conduct a comparative analysis on the Propbank annotation templates completed by the human readers in the machine translation output versus the reference translation.

3.1 Evaluation corpus

The sentences of the evaluation corpus are randomly drawn from the newswire genre of the DARPA GALE program Phase 2.5 evaluation. For each Chinese input sentence, there are one corresponding English reference translation and three state-of-the-art machine translation systems' outputs. The Chinese source and the English reference are annotated with gold standard semantic role labels in Propbank style.

South Korea 's Ministry of Agriculture and Forestry said this evening that an Asian City duck farm reported to the relevant department on the 11th that since the 5th of this month , the number of egg production of over 9,000 ducks in the duck farm had fallen sharply .

Agent 1:	South Korean 's Ministry of Agriculture and Forestry
Action 1:	said
Experiencer1:	an Asian City duck farm reported to the relevant department on the 11th that since the 5th of this month , the number of egg production of over 9,000 ducks in the duck farm had fallen sharply
Temporal 1:	this evening
Agent 2:	an Asan City duck farm
Action 2:	reported
Experiencer 2:	since the 5th of this month , the number of egg production of over 9,000 ducks in the duck farm had fallen sharply
Patient 2:	the relevant department
Temporal 2:	on the 11th
Agent 3:	the number of egg production of over 9,000 ducks in the duck farm
Action 3:	fallen
Temporal 3:	since the 5th of this month
Manner 3:	sharply

Figure 3: Example given to human annotators demonstrating how to label the semantic frames.

Table 1: List of semantic roles that human judges are requested to label.

Label	Event	Label	Event
Actor	who	Temporal	when
Action	did	Location	where
Experiencer	what	Other adverbial arg.	why / how
Patient	whom		

3.2 Reconstruction of semantic frames in MT output

Four groups of bilingual Chinese English human annotators are employed to conduct the analysis. One group of them is given the reference translation. This sanity check serves as the control condition of the analysis. The other three groups of them is given one set of the machine translation system output. The four groups are all disjoint such that no annotators annotate more than one sentence from a MT-reference set to avoid contamination in annotators' judgments. To reduce the effect of personal bias on annotations, each sentence is annotated by at least two human annotators. The results are reported as the average among the annotators.

With the aim of evaluating machine translation utility from a user standpoint, we have simplified the Propbank annotation into a more intuitive event structure, i.e. "who did what to whom, when, where, why and how". Since the layman annotators find that it is difficult to distinguish between the "why" and "how" events type, we have combined the "why" and "how" events in to one "Other adverbial argument" label. Human annotators are given simple and minimal instructions on what to label and two examples demonstrating how to label. Table 1 shows the list of labels annotators are requested to annotate. Figure 3 shows the example shown to the human annotators on how to label semantic frames.

After reconstruction of the semantic frames, the annotated machine translation outputs are distributed to another disjoint group of three monolingual human judges. The human judges are required to match each predicate in the reference translation with those annotated in the MT output. Then, for each matched predicate, they are required to judge whether each of the associated argument in the reference translation is translated and annotated in the MT output: Correct, Incorrect or Partial. Translations of the semantic frames are judged Correct if they express the same meaning as that of the reference translations or the original source input. Translations of the semantic frames are judged Incorrect if they express meaning(s) that belongs in other arguments. Translation of the semantic frames may also be judged Partial if only part of the meaning is correctly expressed. Extra meaning in the semantic frames will not be penalized unless it belongs in another argument. The partially correct category is designed to facilitate a finer-grained measurement of the translation utility.

3.3 SRL based evaluation metric

Based on the comparative matrices collected from the human judges, a precision-recall analysis of accuracy with the reconstructed semantic frames, reflecting the utility of each machine translation system could be done.

$$
\begin{aligned}
C_{\text{core i}} &= \text{no. of Correct core ARG of PRED i in MT} \\
C_{\text{argm i}} &= \text{no. of Correct ARGM of PRED i in MT} \\
P_{\text{core i}} &= \text{no. of Partial core ARG of PRED i in MT} \\
P_{\text{argm i}} &= \text{no. of Partial ARGM of PRED i in MT} \\
\text{MT}_{\text{core i}} &= \text{total no. of core ARG of PRED i in MT} \\
\text{MT}_{\text{argm i}} &= \text{total no. of ARGM of PRED i in MT} \\
\text{Ref}_{\text{core i}} &= \text{total no. of core ARG of PRED i in ref.} \\
\text{Ref}_{\text{argm i}} &= \text{total no. of ARGM of PRED i in ref.}
\end{aligned}
$$

$$
\begin{aligned}
C_{\text{precision}} &= \sum_{\text{all matched}} \frac{w_0 + w_1 C_{\text{core i}} + w_2 C_{\text{argm i}}}{w_0 + w_1 \text{MT}_{\text{core i}} + w_2 \text{MT}_{\text{argm i}}} \\
C_{\text{recall}} &= \sum_{\text{all matched}} \frac{w_0 + w_1 C_{\text{argn i}} + w_2 C_{\text{argmi}}}{w_0 + w_1 \text{Ref}_{\text{core i}} + w_2 \text{Ref}_{\text{argm i}}} \\
P_{\text{precision}} &= \sum_{\text{all matched}} \frac{w_1 P_{\text{core i}} + w_2 P_{\text{argm i}}}{w_0 + w_1 \text{MT}_{\text{core i}} + w_2 \text{MT}_{\text{argm i}}} \\
P_{\text{recall}} &= \sum_{\text{all matched}} \frac{w_1 P_{\text{core i}} + w_2 P_{\text{argm i}}}{w_0 + w_1 \text{Ref}_{\text{core i}} + w_2 \text{Ref}_{\text{argm i}}}
\end{aligned}
$$

$$
\begin{aligned}
\text{Precision} &= \frac{C_{\text{precision}} + (w_{\text{partial}} \times P_{\text{precision}})}{\text{total no. of predicates in MT}} \\
\text{Recall} &= \frac{C_{\text{recall}} + (w_{\text{partial}} \times P_{\text{recall}})}{\text{total no. of predicatesin ref.}} \\
\text{F} - \text{score} &= \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}
\end{aligned}
$$

$C_{\text{core i}}$ and $C_{\text{argm i}}$ represent the number of correctly translated core arguments and adjunct arguments of a matched predicate i respectively while $P_{\text{core i}}$ and $P_{\text{argm i}}$ represent the number of partially translated core arguments and adjunct arguments of a matched predicate i. $\text{MT}_{\text{core i}}$ and $\text{MT}_{\text{argm i}}$ represent the total number of core arguments and adjunct arguments of the matched predicate i in the MT output and $\text{Ref}_{\text{core i}}$ and $\text{Ref}_{\text{argmi}}$ represent the total number of core arguments and adjunct arguments of the matched predicate i in the reference.

$C_{\text{precision}}$ and $P_{\text{precision}}$ are the sum of the portions of correctly or partial correctly translated predicate-argument structures in the MT output. They can be viewed as the true positive for precision. C_{recall} and P_{recall} are the sum of the portion of correctly or partial correctly translated predicate-argument structures in the reference. They can be viewed as the true positive for recall. Note that w_0, w_1 and w_2 are the weights for the matched predicate, core arguments and adjunct arguments. These weights can be viewed as the importance of meanings in the different categories of semantic roles. In this very first preliminary study, we have set them all to 1 and we expect tuning these weights can further increase the correlation of the evaluation metric with human judgment of translation utility.

The precision, recall and f-score of the SRL based MT evaluation metric are defined in terms of the translation accuracy of predicate-argument structures. Note that w_{partial} is the weights for the partially correct translated arguments. In this experiment, we have arbitrarily set it to 0.5.

If all the reconstructed semantic frames in the MT output are completely identical to the gold standard annotation in the reference translation and all the arguments in the reconstructed frames express the same meaning as the corresponding arguments in the reference translations, the f-score of the SRL based MT evaluation metric will be equal to 1.

Table 2: SRL annotation of example 1 MT1 output in figure 1 and the human judgement on translation correctness of each argument.

SRL	reference	MT1	decision
PRED	ceased	–	not match
PRED	resumed	resume	match
ARG0	–	sk - ii the sale of products in the mainland of China	incorrect
ARG1	sales of complete range of SK - II products	sales	partial
TMP	Until after , their sales had ceased in mainland China for almost two months	So far , nearly two months	partial
TMP	now	–	incorrect

3.4 Experiment and Results

Table 2 shows the SRL annotation of MT1 by one of the annotators of example 1 in figure 1

Table 3: SRL based MT evaluation average on all annotators and all sentences.

System	Precision	Recall	F-score
Reference	0.75	0.73	0.73
MT1	0.39	0.35	0.36
MT2	0.37	0.31	0.33
MT3	0.34	0.30	0.30

and the human judgement on translation correctness of each argument. The predicate "ceased" in the reference translation did not match with any predicate annotated in MT1 while the predicate "resumed" matched with the predicate "resume" annotated in MT1. The ARGM-TMP argument, "Until after their sales had ceased in mainland China for almost two months", in the reference translation is partially translated to ARGM-TMP argument, "So far , nearly two months", in MT1; the ARG1 argument, "sales of the complete range of SK - II products", in the reference translation is partially translated to ARG1 argument, "sales", in MT1 and the ARGM-TMP argument, "now" in the reference translation is missing in MT1. The SRL based f-score of this example is 0.33. The final sentence-level SRL based MT evaluation metric of MT1 is the f-score averaged on all annotators. Table 3 shows the results of the SRL based MT evaluation metric averaged on all annotators and all sentences. Our results show that the evaluation metric can successfully distinguish the translation utility of the human translation and the three MT systems; and on system level, MT1 provides the most accurate translation.

4 Inter-annotator Agreement

We measured the inter-annotator agreement in two tasks: role identification and role classification. The standard f-score is used to measure the agreement on SRL annotation as in Brants (2000).

For role identification, the agreement is counted on the matching of word span in the annotated arguments with a tolerance of ± 1 word in mismatch. The tolerance is designed for the fact that annotators are not consistent in handling the articles or punctuations at the beginning or the end of the annotated arguments. The agreement rate on SRL annotations in role identification of reference translation is 76%, and that on MT output is 72%.

For role classification, in addition to the requirement of matching of word span in role identification task, the agreement is counted on the matching of the semantic role labels within two aligned word spans. The agreement rate on SRL annotations of reference translation and that on MT output are 69% and 65% respectively.

The results show that with such minimal training, the layman annotators perform consistently in identifying the semantic structure in both the reference translation and the MT output. The results suggest that the layman annotators also having problem in role confusion and we believe that a slightly more detailed explanation on the role labels may help to clear the confusion.

5 Correlation with human judgments on translation adequacy

We used the Spearman's rank correlation coefficient ρ to measure the correlation of the evaluation metrics with the human judgment on adequacy at sentence-level and took average on the whole data set. The human judgment on adequacy was obtained by showing all three MT outputs together with the Chinese source input to a human reader. The human reader was instructed to order the sentences from the three MT systems according to the accuracy of meaning in the translations. For the MT output, we ranked the sentences from the three MT systems according to the raw scores of the evaluation metrics. The STM scores are calculated based on the syntax tree of the reference and MT output parsed by the Charniak parser (Charniak, 2001). Table 4 shows the raw scores of example 1 under the our proposed SRL based evaluation metric, sentence-level BLEU, sentence-level STM and the corresponding ranks assigned to each of the systems, together with the human ranks on adequacy.

The Spearman's rank correlation coefficient ρ can be calculated using the following simplified equation:

$$\rho = 1 - \frac{6 \sum {d_i}^2}{n(n^2 - 1)}$$

where d_i is the difference between the ranks of the

Table 4: Sentence-level SRL based f-score evaluation metrics average on annotators, sentence-level BLEU, sentence-level STM, their corresponding rank assigned and the human rank on adequacy for example 1.

System	MT output	SRL		BLEU		STM		Human
		score	rank	score	rank	score	rank	rank
Src	至此， 在 中国 内地 停售 了 近 两 个 月 的 ＳＫ－ＩＩ 全线 产品 恢复 销售 。	-	-	-	-	-	-	-
Ref	Until after their sales had ceased in mainland China for almost two months , sales of the complete range of SK － II products have now be resumed .	-	-	-	-	-	-	-
MT1	So far , nearly two months sk - ii the sale of products in the mainland of China to resume sales .	0.167	2	0.012	3	0.364	1	2
MT2	So far , in the mainland of China to stop selling nearly two months of SK - 2 products sales resumed .	0.317	1	0.013	2	0.303	3	1
MT3	So far , the sale in the mainland of China for nearly two months of SK - II line of products .	0.000	3	0.124	1	0.344	2	3

Table 5: Average sentence-level correlation for the evaluation metrics.

Metric	Correlation with human
SRL based evaluation	0.41
BLEU	0.25
STM	0.32

evaluation metrics and the human judgment over of system i and n is the number of systems. The range of possible values of correlation coefficient is [-1,1], where 1 means the systems are ranked in the same order as the human judgment and -1 means the systems are ranked in the reverse order as the human judgment. The higher the value for ρ indicates the more similar the ranking by the evaluation metric to the human judgment.

Our results show that the proposed SRL based evaluation metric has a higher correlation with the human judgment on adequacy than either the BLEU or STM metrics. Table 5 compares the average sentence-level ρ for our proposed SRL based evaluation metric, BLEU, and STM. The correlation coefficient for the proposed SRL based evaluation metric is 0.41, while that for BLEU is 0.25. The correlation coefficient for STM is 0.32, significantly better than BLEU, but still far short of our SRL based metric.

6 Conclusions and Future Work

We presented results of an empirical study on evaluation the utility of MT output, by assessing the accuracy with which human reader are able to complete the SRL templates. The SRL based f-score evaluation metric we proposed provided an intuitive picture on how much information of the original source input the machine translation users can extract by reading the MT output. Comparing to HTER where the human decision is heavy and requires advance knowledge in how to fix the translation with minimum change, only minimal instructions is necessary to be given to the human readers in our proposed metric. The human readers may not necessarily be translation experts.

Our results show that using SRL in semantic MT evaluation is a highly promising direction for further research. We evaluated the proposed SRL based metric with human judgment on adequacy using Spearman's correlation coefficient. The proposed SRL based evaluation metric was found to be significantly better correlated to human judgment on adequacy than either BLEU or the syntax-based evaluation metric STM.

Our current direction is to discriminatively tune the weights within the SRL based evaluation metric, so as to further increase the correlation of the metric with human judgment.

Our other main avenue of current work is to construct automated metrics approximating the

evaluation method described here (which provides an upper bound for automated SRL-based metrics). With the improving performance of shallow semantic parsers, we believe that the proposed evaluation metric could be further developed into inexpensive automatic MT evaluation metrics.

7 Acknowledgments

This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA) under GALE Contract Nos. HR0011-06-C-0022 and HR0011-06-C-0023 and by the Hong Kong Research Grants Council (RGC) research grants GRF621008, GRF612806, DAG03/04.EG09, RGC6256/00E, and RGC6083/99E. 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 Defense Advanced Research Projects Agency.

References

T. Brants. Inter-annotator agreement for a German newspaper corpus. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC-2000)*. Citeseer, 2000.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of BLEU in machine translation research. In *Proceedings of EACL 2006*, pages 249–256, 2006.

Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. (Meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158. Association for Computational Linguistics, 2007.

Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. Further meta-evaluation of machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 70–106. Association for Computational Linguistics, 2008.

Marine Carpuat and Dekai Wu. Improving Statistical Machine Translation using Word Sense Disambiguation. In *Conference on Empirical Methods in Natural Language Processing and Conference on Computational Natural Language Learning (EMNLP-CoNLL 2007)*, Prague, Jun 2007.

Y.S. Chan, H.T. Ng, and D. Chiang. Word sense disambiguation improves statistical machine translation. In *45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech, 2007.

E. Charniak. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, page 131. Association for Computational Linguistics, 2001.

Pascale Fung, Zhaojun Wu, Yongsheng Yang, and Dekai Wu. Automatic Learning of Chinese English Semantic Structure Mapping. In *IEEE Spoken Language Technology Workshop, 2006*, pages 230–233, 2006.

Jesús Giménez and Lluís Màrquez. Discriminative Phrase Selection for Statistical Machine Translation. *Learning Machine Translation*, 2007.

Jesús Giménez and Lluís Màrquez. Linguistic features for automatic evaluation of heterogenous MT systems. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 256–264, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

Jesús Giménez and Lluís Màrquez. A smorgasbord of features for automatic MT evaluation. In *Proceedings of the 3rd Workshop on Statistical Machine Translation*, pages 195–198, Columbus, OH, June 2008. Association for Computational Linguistics.

Philipp Koehn and Christof Monz. Manual and automatic evaluation of machine translation between european languages. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 102–121. Association for Computational Linguistics, 2006.

D. Liu and D. Gildea. Syntactic features for evaluation of machine translation. *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, page 25, 2005.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: an Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1):71–106, 2005.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James H. Martin, and Dan Jurafsky. Shallow Semantic Parsing Using Support Vector Machines. In *Proceedings of NAACL-HLT 2004*, 2004.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, 2006.

Dekai Wu and Pascale Fung. Can Semantic Role Labeling Improve SMT? In *Proceedings of the 13th Annual Conference of the EAMT*, pages 218–225, Barcelona, Spain, May 2009.

Dekai Wu and Pascale Fung. Semantic Roles for SMT: A Hybrid Two-Pass Model. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 13–16. Association for Computational Linguistics, 2009.

Nianwen Xue and Martha Palmer. Automatic Semantic Role Labeling for Chinese Verbs. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence, Edinburgh, Scotland*, 2005.

Arabic morpho-syntactic feature disambiguation in a translation context

Ines Turki Khemakhem, Salma Jamoussi, Abdelmajid Ben Hamadou

MIRACL Laboratory, ISIM Sfax, Pôle Technologique

ines_turki@yahoo.fr, salma.jammoussi@isimsf.rnu.tn, abdelmajid.benhamadou@isimsf.rnu.tn

Abstract

Morphological analysis and disambiguation are crucial stages in a variety of natural language processing applications such as machine translation, especially when languages with complex morphology are concerned such as Arabic. Arabic is a highly flexional language, in that, the same root can lead to various forms according to its context. In this paper, we present a system which disambiguates the output of a morphological analyzer for Arabic. The Arabic morphological analyzer used consists of a set of all possible morphological analyses for each word, with the unique correct syntactic feature. We want to choose the correct features using the features generated by the morphological analyzer for the French language in the other side. To obtain this data, we used the results of the alignment of word trained with GIZA++ (Och and Ney, 2003).

1 Introduction

Arabic is characterized by a rich morphology. Due to the fact that the Arabic script usually does not encode short vowels, the degree of morphological ambiguity is very high. In addition to being inflected for gender, number, words can be attached to various clitics for conjunction "و" (and), the definite article "ال" (the), prepositions (e.g. "ب" (by/with), "ل" (for), "ك" (as)), object pronouns (e.g. "هم" (their/them)).

The morphological analysis of a word consists of determining morphological information about each word, such as part-of-speech (i.e., noun, verb, particle, etc), voice, gender, number, information about the clitics, etc. Morphological analysis and disambiguation are crucial preprocessing steps for a variety of natural language processing applications, from search and information extraction to machine translation. For languages with complex morphology these are nontrivial processes.

Arabic words are often ambiguous in their morphological analysis. This is due to Arabic's rich system of affixation and clitics and the omission of disambiguating short vowels. The problem is that many words have different meanings depending on their diacritization. This leads to ambiguity when processing data for natural language processing applications such as machine translation. This propriety has an important implication for statistical modeling of the Arabic language.

In this paper, we present a novel morphology preprocessing technique for Arabic. We exploit the Arabic morphology-French alignment to choose a correct morpho-syntactic feature produced by a morphological analyzer.

This paper is organized as follows: section 2 gives a brief description of some related works to the introduction of morphological disambiguation. Section 3 presents the used morphological analyzer Morph2 for Arabic texts, able to recognize word composition and to provide more specific morphological information about it. We present in section 4 some problems in context morpho-syntactic feature choice; in the remainder of this section we discuss the complexity of Arabic morphology and the challenge of morphological disambiguation. Section 5 gives a short overview of the data and tools used for our Arabic word Morpho-syntactic feature disambiguation and shows the experimental details of our system. Finally, section 6 presents some conclusions.

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 61–65,
COLING 2010, Beijing, August 2010.

2 Related work

Morphological analysis and disambiguation are crucial pre-processing steps for a variety of natural language processing applications.

Previous research has focused on disambiguating the output of a morphological analyzer. Hajic (2000) is the first to use a dictionary as a source of possible morphological analyses (and hence tags) for an inflected word form. He convincingly shows that for five Eastern European languages with complex inflection plus English, using a morphological analyzer improves performance of a tagger. He concludes that for highly inflectional languages "the use of an independent morphological dictionary is the preferred choice more annotated data". He redefines the tagging task as a choice among the tags proposed by the dictionary, using a log-linear model trained on specific ambiguity classes for individual morphological features. Hajic (2000) demonstrates convincingly that morphological disambiguation can be aided by a morphological analyzer, which, given a word without any context, gives us the set of all possible morphological tags.

The only work on Arabic tagging that uses a corpus for training and evaluation, (Diab et al., 2004), does not use a morphological analyzer. Diab et al. (2004) perform tokenization, POS tagging and base phrase chunking using a SVM based learner.

The Morphological Analysis and Disambiguation of Arabic (MADA) system is described in (Habash and Rambow, 2005). The basic approach used in MADA is inspired by the work of Hajic (2000) for tagging morphologically rich languages, which was extended to Arabic. Habash and Rambow (2005) use SVM-classifiers for individual morphological features and a simple combining scheme for choosing among competing analyses proposed by the dictionary.

3 Arabic word segmenter

Arabic is a morphologically complex language. Compared with French, an Arabic word can sometimes correspond to a whole French sentence (Example : the Arabic word "أتتذكروننا" corresponds in French to the sentence "Est-ce que vous vous souvenez de nous", in English: "Do you remember us").

The aim of a morphological analysis step is to recognize word composition and to provide specific morphological information about it. For Example : the word "يعرفون" (in French: connaîssent, in English: they know) is the result of the concatenation of the prefix "ي" indicating the present and suffix "ون" indicating the plural masculine of the verb "عرف" (in French: connaître, in English: to know). The morphological analyzer determines for each word the list of all its possible morphological features.

In Arabic language, some conjugated verbs or inflected nouns can have the same orthographic form due to absence of vowels (Example : non-voweled Arabic word "فصل" can be a verb in the past "فَصَلَ" (He dismissed), or a masculine noun "فَصْلٌ" (chapter / season), or a concatenation of the coordinating conjunction "فَ " (then) with the verb "صل": imperative of the verb (bind)).

In this work, In order to handle the morphological ambiguities, we decide to use MORPH2 (Belguith et al., 2006), an Arabic morphological analyzer developed at the Miracl laboratory[1]. MORPH2 is based on a knowledge-based computational method. It accepts as input an Arabic text, a sentence or a word. Its morphological disambiguation and analysis method is based on five steps:

- A tokenization process is applied in a first step. It consists of two sub-steps. First, the text is divided into sentences, using the system Star (Belguith et al., 2005), an Arabic text tokenizer based on contextual exploration of punctuation marks and conjunctions of coordination. The second sub-step detects the different words in each sentence.
- A morphological preprocessing step which aims to extract clitics agglutinated to the word. A filtering process is then applied to check out if the remaining word is a particle, a number, a date, or a proper noun.
- An affixal analysis is then applied to determine all possible affixes and roots. It aims to identify basic elements belonging

[1] http://www.miracl.rnu.tn

to the constitution of a word (the root and affixes i.e. prefix, infix and suffix).

- The morphological analysis step consists of determining for each word, all its possible morpho-syntactic features (i.e, part of speech, gender, number, time, person, etc.). Morpho-syntactic features detection is made up on three stages. The first stage identifies the part-of-speech of the word (i.e. verb "فعل", noun "اسم", particle "أداة" and proper noun "اسم علم"). The second stage extracts for each part-of-speech a list of its morpho-syntactic features. A filtering of these feature lists is made in the third stage.
- Vocalization and validation step : each handled word is fully vocalized according to its morpho-syntactic features determined in the previous step.

In our method, each Arabic word, from Arabic data, is replaced by its segmented form, where stem, clitic and affix are featured with their morphological classes (e.g. proclitic, prefix, stem, suffix and enclitic). For example: the word "فعرفناهم" (in French: "et nous les avons connu", in English: "and we have known them") is the result of the concatenation of the proclitic "فَ" (then): coordinating conjunction, the suffix "نا" for the present masculine plural, enclitic "هم" (for the masculine plural possession pronoun), and the rest of the word "عرف" indicating the stem. So, the word "فعرفناهم" will be replaced by:

"enclitic_ هم suffix_نا **Stem_**عرف proclitic_ف"

4 Problems in context Morpho-syntactic feature choice

As mentioned in section 1, ambiguities in Arabic word are mainly caused by the absence of the short vowels. Thus, a word can have different meanings. There are also the usual homographs of uninflected words with/without the same pronunciation, which have different meanings and usually different POS's. For example: the word "ذهب", in English: "gold" and "ذهب", in English: "go". In Arabic there are four categories of words: noun, proper noun, verbs and particles. The absence of short vowels can cause ambiguities within the same category or across different categories. For example: the word "بعد" corresponds to many categories (table 1).

meanings of a word "بعد"	Categories
after	Particule
remoteness	Noun
remove	Verb
go away	Verb

Table 1. Different meanings of a word "بعد"

Arabic uses a diverse system of prefixes, suffixes, and pronouns that are attached to the words (Soudi, 2007).

In fact, in Arabic language, the word: "وضع" can be a verb in the past (He filed), or a masculine noun (state), or a concatenation of the coordinating conjunction "و" (and) with the verb "ضع": imperative of the verb (filed). For this reason, correct morphological analysis is required to resolve structural ambiguities among Arabic sentence.

5 Word Morpho-syntactic feature disambiguation

5.1 Training corpus

The training corpus used in this work is an Arabic-French bitext aligned at the sentence level. Each Arabic word, from Arabic data, is replaced by its segmented form. In the other side, the French corpus is part-of-speech (POS) tagged by using treetagger tool (Schmid, 1994) for annotating text with part-of-speech and lemma information.

5.2 Alignment model

The aligned model was trained with GIZA++ (Och and Ney, 2003), which implements the most typical IBM and HMM alignment models. The alignment model used consists of IBM-1, HMM, IBM-3 and IBM-4.

5.3 Using treetagger for Arabic Word Morpho-syntactic feature disambiguation

To pre-process the Arabic data, we use the MORPH2 morphological analyzer (Belguith et al., 2006). A sample output of the morphological analyzer is shown in Figure 1.

Figure 1. Possible analyses for the word "بعد"

The obtained output consists of a set of all possible morphological analyses for each word, with the unique correct analysis. One needs to select the right meaning by looking at the context. Given the highly inflection nature of Arabic, resolving ambiguities is syntactically harder within the same category. We want to choose the correct output using the features generated by TreeTagger applied to the French corpus.

To obtain this correct feature, we needed to match data in the segmented Arabic corpus to the lexeme and feature representation output by TreeTagger. The matching included the results of the alignment of word between the segmented Arabic corpus and the part-of-speech tagged French corpus.

Example : the word "فعرفناهم" (in French: "et nous les avons connu", in English: "and we have known them") is segmented by:

"enclitic_هم suffix_نا **stem**_عرف proclitic_ف"

The part-of-speech tagged of the French sentence is:

"et_KON nous_PRO:PER les_PRO:PER avons_VER:pres connu_VER:pper"

The result of the alignment between these two sentences is:

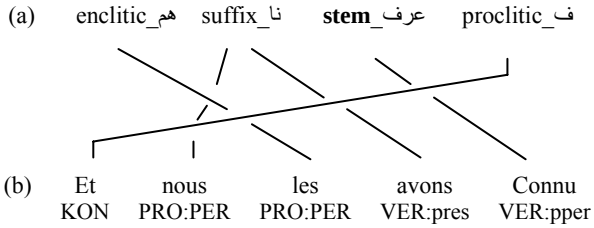


Figure 2. (a) Output of morphological analysis MORPH2: segmented Arabic sentence, (b) French translation and its alignment with segmented morphological analysis

The stem "عرف" is aligned with the word: "connu" : the past participle of the verb "connaître" (in English : "known"). We can deduce that the part of speech of the stem "عرف" is a verb.

Morpho-syntactic features provided by TreeTagger are verb, proper noun, noun, adjective, adverb, conjunction, pronoun, preposition, etc. The morpho-syntactic feature of Arabic words aligned with French words tagged by adjective or noun will be replaced by the morpho-syntactic feature : noun: "اسم". While, the morpho-syntactic feature : adverb, conjunction, or prepostion will be replaced by the Arabic morpho-syntactic feature : particle : "أداة".

We can attest that the use of morpho-syntactic features provided by the part-of-speech tagged corpus in the other side can remove disambiguation of morpho-syntactic feature of the Arabic word provided by a morphological analyser, especially for agglutinative and inflectional languages.

5.4 Experimental results

In our experiments, on the entire corpus, the MORPH2 morphological analyzer makes 1152 errors (27%). Table 2 shows the results obtained with the morphological analyzer MORPH2, BASELINE, and the results obtained with the Arabic morphology-French alignment, treetagger-to-morph2, where Arabic morphology-French alignment is used to choose a correct

morpho-syntactic feature produced by a morphological analyzer MORPH2.

System	Accuracy (%)
BASELINE	73%
treetagger-to-morph2	88%

Table 2. Results of treetagger-to-morph2 compared against BASELINE on the task of POS tagging of Arabic text

Thus one can observe that, for Arabic, the treetagger-to-morph2 outperforms the BASELINE tagger with a significant absolute difference of 15% in tagging accuracy.

The performance of treetagger-to-morph2 is better than the baseline BASELINE. The errors encountered result from confusing nouns with verbs, particles or vice versa. This is to be caused by the presence of homographs of Arabic words, which have different meanings and different POS's.

6 Conclusion

Morphological disambiguation of Arabic is a difficult task which involves, in theory, thousands of possible tags.

In this paper, we present a system which disambiguates the output of a morphological analyzer for Arabic. Arabic is a morphologically rich language, and Morphological analysis and disambiguation are crucial stages in a variety of natural language processing applications.

We first applied an Arabic word segmentation step, to improve the alignments models. So we use the Arabic morphological analyzer MORPH2. Then, we proposed to use TreeTagger tool, able to annotate text with part-of-speech and lemma information, where each word from French corpus is agglutinated to its part-of-speech (POS). The sentence alignment between Arabic and French corpus was trained with GIZA++. The core idea is to avoid morpho-syntactic ambiguity of the Arabic words obtained in the MORPH2 output by using the part of speech of corresponding aligned French word. We showed that imposing Arabic-French alignment dependent constraints on possible sequences of analyses improves the morphological disambiguation.

Future work will focus on taking advantage of our efficient technique. We are very interested to use Word Morpho-syntactic feature disambiguation to build up an efficient French-Arabic translation system.

References

Belguith L., and Chaâben N. 2006. Analyse et désambiguïsation morphologiques de textes arabes non voyellés, *Actes de la 13ème confrence sur le Traitement Automatique des Langues Naturelles*, Leuven Belgique, 493-501.

Belguith L., Baccour L. and Mourad G. 2005. Segmentation des textes arabes basée sur l'analyse contextuelle des signes de ponctuations et de certaines particules. *Actes de la 12éme Conférence annuelle sur le Traitement Automatique des Langues Naturelles,* 451-456.

Diab M., Hacioglu K., and Jurafsky D. 2004. Automatic tagging of Arabic text: From raw text to base phrase chunks. *In 5th Meeting of the North American Chapter of the Association for Computational Linguistics/Human Language Technologies Conference (HLT-NAACL04),* Boston, MA.

Habash N. and Rambow O.. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), Ann Arbor, Michigan, June. Association for Computational Linguistics. 573–580

Hajic J. 2000. Morphological tagging: Data vs. dictionaries. *In 1st Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL'00)*, Seattle, WA.

Och F. J., and Ney H. 2003. A Systematic comparison of various statistical alignment models, *Computational Linguistics*, 29(1): 19-51.

Och F. J., Ney H. 2004. The alignment template approach to statistical machine translation, *Computational Linguistics*, 30(4): 417-449.

Schmid H. 1994. Probabilistic Part-of-speech Tagging Using Decision Trees. *In Proceedings of International Conference on New Methods in Language Processing*, Manchester. 44-49.

Soudi A., Bosch A. and Neumann G. Arabic Computational Morphology: Knowledge-based and Empirical Methods. Springer, 2007.

A Discriminative Approach for Dependency Based Statistical Machine Translation

Sriram Venkatapathy
LTRC, IIIT-Hyderabad
sriram@research.iiit.ac.in

Rajeev Sangal
LTRC, IIIT-Hyderabad
sangal@mail.iiit.ac.in

Aravind Joshi
University of Pennsylvania
joshi@seas.upenn.edu

Karthik Gali[1]
Talentica
karthik.gali@gmail.com

Abstract

In this paper, we propose a dependency based statistical system that uses discriminative techniques to train its parameters. We conducted experiments on an English-Hindi parallel corpora. The use of syntax (dependency tree) allows us to address the large word-reorderings between English and Hindi. And, discriminative training allows us to use rich feature sets, including linguistic features that are useful in the machine translation task. We present results of the experimental implementation of the system in this paper.

1 Introduction

Syntax based approaches for Machine Translation (MT) have gained popularity in recent times because of their ability to handle long distance reorderings (Wu, 1997; Yamada and Knight, 2002; Quirk et al., 2005; Chiang, 2005), especially for divergent language pairs such as English-Hindi (or English-Urdu). Languages such as Hindi are also known for their rich morphology and long distance agreement of features of syntactically related units. The morphological richness can be handled by employing techniques that factor the lexical items into morphological factors. This strategy is also useful in the context of English-Hindi MT (Bharati et al., 1997; Bharati et al., 2002; Ananthakrishnan et al., 2008; Ramanathan et al., 2009) where there is very limited parallel corpora available, and breaking words into smaller units helps in reducing sparsity. In order to handle phenomenon such as long-distance word agreement to achieve accurate generation of target language words, the inter-dependence between the factors of syntactically related words need to be modelled effectively.

Some of the limitations with the syntax based approaches such as (Yamada and Knight, 2002; Quirk et al., 2005; Chiang, 2005) are, (1) They do not offer flexibility for adding linguistically motivated features, and (2) It is not possible to use morphological factors in the syntax based approaches. In a recent work (Shen et al., 2009), linguistic and contextual information was effectively used in the framework of a hierarchical machine translation system. In their work, four linguistic and contextual features are used for accurate selection of translation rules. In our approach in contrast, linguistically motivated features can be defined that directly effect the prediction of various elements in the target during the translation process. This features use syntactic labels and collocation statistics in order to allow effective training of the model.

Some of the other approaches related to our model are the Direct Translation Model 2 (DTM2) (Ittycheriah and Roukos, 2007), End-to-End Discriminative Approach to MT (Liang et al., 2006) and Factored Translation Models (Koehn and Hoang, 2007). In DTM2, a discriminative trans-

[1]This work was done at LTRC, IIIT-Hyderabad, when he was a masters student, till July 2008

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 66–74,
COLING 2010, Beijing, August 2010.

lation model is defined in the setting of a phrase based translation system. In their approach, the features are optimized globally. In contrast to their approach, we define a discriminative model for translation in the setting of a syntax based machine translation system. This allows us to use both the power of a syntax based approach, as well as, the power of a large feature space during translation. In our approach, the weights are optimized in order to achieve an accurate prediction of the individual target nodes, and their relative positions.

We propose an approach for syntax based statistical machine translation which models the following aspects of language divergence effectively.

- *Word-order variation* including long-distance reordering which is prevalent between language pairs such as English-Hindi and English-Japanese.
- *Generation of word-forms* in the target language by predicting the word and its factors. During prediction, the *inter-dependence of factors* of the target word form with the factors of syntactically related words is considered.

To accomplish this goal, we visualize the problem of MT as transformation from a morphologically analyzed source syntactic structure to a target syntactic structure[1] (See Figure 1). The transformation is factorized into a series of mini-transformations, which we address as features of the transformation. The features denote the various linguistic modifications in the source structure to obtain the target syntactic structure. Some of the examples of features are lexical translation of a particular source node, the ordering at a particular source node etc. These features can be entirely local to a particular node in the syntactic structure or can span across syntactically related entities. More about the features (or mini-transformations) is explained in section 3. The transformation of a source syntactic structure is scored by taking a weighted sum of its features [2]. Let τ represent the transformation of source syntactic structure s, the score of transformation is computed as represented in Equation 1.

$$score(\tau|s) = \sum_i w_i * f_i(\tau, s) \qquad (1)$$

In Equation 1, $f_i's$ are the various features of transformation and $w_i's$ are the weights of the features. The strength of our approach lies in the flexibility it offers in incorporating linguistic features that are useful in the task of machine translation. These features are also known as *prediction features* as they map from source language information to information in the target language that is being predicted.

During decoding a source sentence, the goal is to choose a transformation that has the highest score. The source syntactic structure is traversed in a bottom-up fashion and the target syntactic structure is simultaneously built. We used a bottom-up traversal while decoding because it builds a contiguous sequence of nodes for the subtrees during traversal enabling the application of a wide variety of language models.

In the training phase, the task is to learn the weights of features. We use an online large-margin training algorithm, MIRA (Crammer et al., 2005), for learning the weights. The weights are locally updated at every source node during the bottom-up traversal of the source structure. For training the translation model, automatically obtained word-aligned parallel corpus is used. We used GIZA++ (Och and Ney, 2003) along with the *growing* heuristics to word-align the training corpus.

The basic factors of the word used in our experiments are root, part-of-speech, gender, number and person. In Hindi, common nouns and verbs have gender information whereas, English doesn't contain that information. Apart from the basic factors, we also consider the role information provided by labelled dependency parsers. For computing the dependency tree on the source side, We used stanford parser (Klein and Manning, 2003) in the experiments presented in this chapter[3].

[1]Note that target structure contains only the target factors. An accurate and deterministic morphological generator combines these factors to produce the target word form.

[2]The features can be either binary-values or real-valued

[3]Stanford parser gives both the phrase-structure tree as well as dependency relations for a sentence.

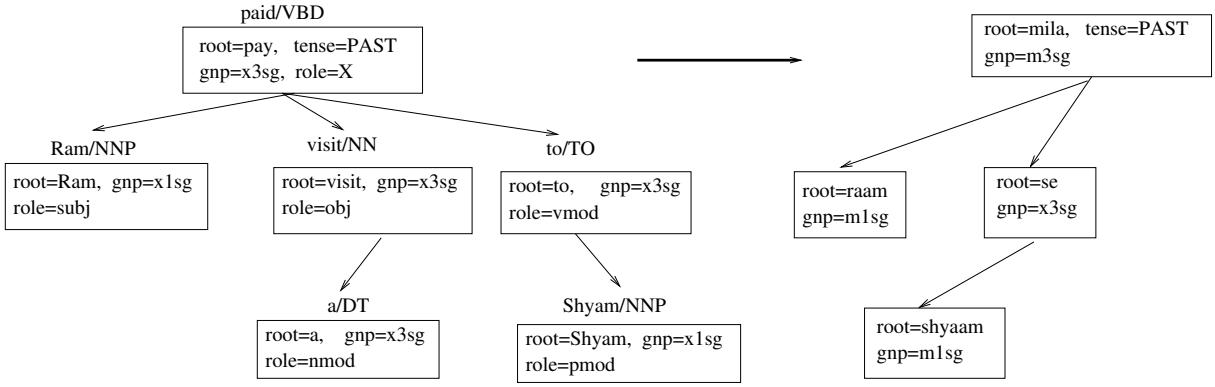


Figure 1: Transformation from source structure to target language

The function words such as prepositions and auxiliary verbs largely express the grammatical roles/functions of the content words in the sentence. In fact, in many agglutinative languages, these words are commonly attached to the content word to form one word form. In this paper, we also conduct experiments where we begin by grouping the function words with their corresponding function words. These groups of words are called local-word groups. In these cases, the function words are considered as factors of the content words. Section 2 explains more about the local word groups in English and Hindi.

2 Local Word Groups

Local word groups (LWGs) (Bharati et al., 1998; Vaidya et al., 2009) consist of a content word and its associated function words. Local word grouping reduces a sentence to a sequence of content words with the case-markers and tense-markers acting as their factors. For example, consider an English sentence 'People of these island have adopted Hindi as a means of communication'. 'have adopted' is a LWG with root 'adopt' and tense markers being 'have_ed'. Another example for the LWG will be 'of communication' where 'communication' is the root, and 'of' is the case-marker. It is to be noted that Local word grouping is different from chunking, where more than one content word can be part of a chunk. We obtain local word groups in English by processing the output of the stanford parser. In Hindi, the function words always appear immediately after the content word[4], and it requires simple pattern

matching to obtain the LWGs. The rules applied are, (1) VM (RB|VAUX)+, and (2) N.* IN.

3 Features

There are three types of transformation features explored by us, (1) Local Features, (2) Syntactic Features and, (3) Contextual Features. In this section, we describe each of these categories of features representing different aspects of transformation with examples.

3.1 Local Features

The local features capture aspects of local transformation of an *atomic treelet* in the source structure to an atomic treelet in the target language. Atomic treelet is a semantically non-decomposible group of one or more nodes in the syntactic structure. It usually contains only one node, except for the case of multi-word expressions (MWEs). Figure 2 presents the examples of local transformation.

Some of the local features used by us in our experiments are (1) *dice coefficient*, (2) dice coefficient of roots, (3) dice coefficient of null translations, (4) *treelet translation probability*, (5) *gnp-gnp pair*, (5) *preposition-postposition pair*, (6) *tense-tense pair*, (7) *part-of-speech fertility* etc. *Dice coefficients* and *treelet translation probabilities* are measures that express the statistical co-occurrence of the atomic treelets.

[4]case-markers are called postpositions

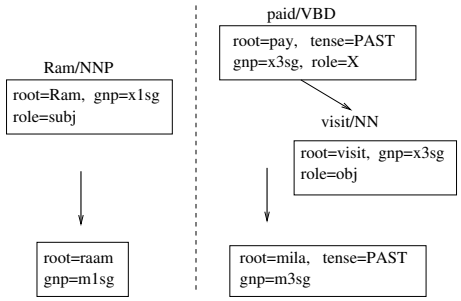


Figure 2: Local transformations

3.2 Syntactic Features

The syntactic features are used to model the difference in the word orders of the two languages. At every node of the source syntactic structure, these features define the changes in the relative order of children during the process of transformation. They heavily use source information such as part-of-speech tags and syntactic roles of the source nodes. One of the features used is *reorderPostags*.

This feature captures the change in relative positions of children with respect to their parents during the tree transformation. An example feature for the transformation given in Figure 1 is shown in Figure 3.

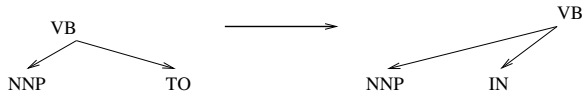


Figure 3: Syntactic feature - reorder postags

The feature *reorderPostags* is in the form of a complete transfer rule. To handle cases, where the left-hand side of 'reorderPostags' does not match the syntactic structure of the source tree, the simpler feature functions are used to qualify various reorderings. Instead of using POS tags, feature functions can be defined that use syntactic roles.

Apart from the above feature functions, we can also have features that compute the score of a particular order of children using syntactic language models (Gali and Venkatapathy, 2009; Guo et al., 2008). Different features can be defined that use different levels of information pertaining to the atomic treelet and its children.

3.3 Contextual Features

Contextual features model the inter-dependence of factors of nodes connected by dependency arcs. These features are used to enable access to global information for prediction of target nodes (words and its factors).

One of the features *diceCoeffParent*, relates the parent of a source node to the corresponding target node (see figure 4.

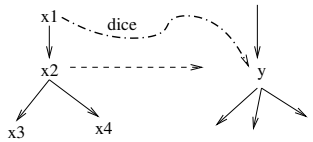


Figure 4: Use of Contextual (parent) information of $x2$ for generation of y

The use of this feature is expected to address of the limitations of using 'atomic treelets' as the basic units in contrast to phrase based systems which consider arbitrary sequences of words as units to encode the local contextual information. In my case, We relate the target treelet with the contextual information of the source treelet using feature functions rather than using larger units. Similar features are used to connect the context of a source node to the target node.

Various feature functions are defined to handle interaction between the factors of syntactically related treelets. The gender-number-person agreement is a factor that is dependent of gender-number-person factors of the syntactically related treelets in Hindi. The rules being learnt here are simple. However, more complex interactions can also be handled though features such as *prep_Tense* where, the case-marker in the target is linked to the tense of parent verb.

4 Decoding

The goal is to compute the most probable target sentence given a source sentence. First, the source sentence is analyzed using a morphological analyzer[5], local word grouper (see section 2) and a dependency parser. Given the source structure, the task of the decoding algorithm is to choose the transformation that has the maximum score.

[5]http://www.cis.upenn.edu/∼xtag/

The dependency tree of the source language sentence is traversed in a bottom-up fashion for building the target language structure. At every source node during the traversal, the local transformation is first computed. Then, the relative order of its children is then computed using the syntactic features. This results in a target structure associated with the subtree rooted at the particular node. The target structure associated with the root node of the source structure is the result of the best transformation of the entire source structure.

Hence, the task of computing the best transformation of the entire source structure is factorized into the tasks of computing the best transformations of the source treelets. The equation for computing the score of a transformation, Equation 1, can be modified as Equation 2 given below.

$$score(\tau|s) = \sum_{r} |r| * \sum_{i} w_i * f_i(\tau_r, r) \quad (2)$$

where, τ_j is the local transformation of the source treelet r. The best transformation $\hat{\tau}$ of source sentence s is,

$$\hat{\tau} \quad = \quad argmax_{\tau}\ score(\tau|s) \quad (3)$$

5 Training Algorithm

The goal of the training algorithm is to learn the feature weights from the word aligned corpus. For word-alignment, we used the IBM Model 5 implemented in GIZA++ along with the *growing* heuristics (Koehn et al., 2003). The gold atomic treelets in the source and their transformation is obtained by mapping the source node to the target using the word-alignment information. This information is stored in the form of transformation tables that is used for the prediction of target atomic treelets, prepositions and other factors. The transformation tables are pruned in order to limit the search and eliminate redundant information. For each source element, only the top few entries are retained in the table. This limit ranges from 3 to 20.

We used an online-large margin algorithm, MIRA (McDonald and Pereira, 2006; Crammer et al., 2005), for updating the weights. During parameter optimization, it is sometimes impossible to achieve the gold transformation for a node because the pruned transformation tables may not lead to the target gold prediction for the source node. In such cases where the gold transformation is unreachable, the weights are not updated at all for the source node as it might cause erroneous weight updates. We conducted our experiments by considering both the cases, (1) Identifying source nodes with unreachable transformations, and (2) Updating weights for all the source nodes (till a maximum iteration limit). The number of iterations on the entire corpus can also be fixed. Typically, two iterations have been found to be sufficient to train the model.

The dependency tree is traversed in a bottom-up fashion and the weights are updated at each source node.

6 Experiments and Results

The important aspects of the translation model proposed in this paper have been implemented. Some of the components that handle word insertions and non-projective transformations have not yet been implemented in the decoder, and should be considered beyond the scope of this paper. The focus of this work has been to build a working syntax based statistical machine translation system, which can act as a platform for further experiments on similar lines. The system would be available for download at http://shakti.iiit.ac.in/∼sriram/vaanee.html. To evaluate this experimental system, a restricted set of experiments are conducted. The experiments are conducted on the English-Hindi language pair using a corpus in tourism domain containing 11300 sentence pairs[6].

6.1 Training

6.1.1 Configuration

For training, we used DIT-TOURISM-ALIGN-TRAIN dataset which is the word-aligned dataset of 11300 sentence pairs. The word-alignment is done using GIZA++ (Och and Ney, 2003) toolkit and then *growing* heuristics are applied. For our experiments, we use two *growing* heuristics, GROW-DIAG-FINAL-AND and GROW-DIAG-FINAL as they cover most number of words in both the sides of the parallel corpora.

[6]DIT-TOURISM corpus

Number of Training Sentences	500
Iterations on Corpus	1-2
Parameter optimization algorithm	MIRA
Beam Size	1-20
Maximum update attempts at source node	1-4
Unreachable updates	False
Size of transformation tables	3

Table 1: Training Configuration

The training of the model can be performed under different configurations. The configurations that we used for the training experiments are given in Table 6.1.1.

6.2 Results

For the complete training, the number of sentences that should be used for the best performance of the decoder should be the complete set. In the paper, we have conducted experiments by considering 500 training sentences to observe the best training configuration.

At a source node, the weight vector is iteratively updated till the system predicts the gold transformation. We conducted experiments by fixing the maximum number of update attempts. A source node, where the gold transformation is not achieved even after the maximum updates limit, the update at this source node is termed a *update failure*. The source nodes, where the gold transformation is achieved even without making any updates is known as the *correct prediction*.

At some of the source nodes, it is not possible to arrive at the gold target transformation because of limited size of the training corpus. At such nodes, we have avoided doing any weight update. As the desired transformation is unachievable, any attempt to update the weight vector would cause noisy weight updates.

We observe various parameters to check the effectiveness of the training configuration. One of the parameters (which we refer to as 'updateHits') computes the number of successful updates (S) performed at the source nodes in contrast to number of failed updates (F). Successful updates result in the prediction of the transformation that is same as the reference transformation. A failed update doesn't result in the achievement of the correct prediction even after the maximum iteration limit (see section 6.1.1) is reached. At some of the source nodes, the reference transformations are *unreachable* (U). The goal is to choose the configuration that has least number of average failed updates (F) because it implies that the model has been learnt effectively.

			UpdateHit			
	K	**m**	P	S	F	U
1.	1	4	1680	2692	84	4081
2.	5	4	1595	2786	75	4081
3.	10	4	1608	2799	49	4081
4.	20	4	1610	2799	**47**	4081

Table 2: Training Statistics - Effect of Beam Size

From Table 2, we can see that the bigger beam size leads to a better training of the model. The beam size was varied between 1 and 20, and the number of update failures (F) was observed to be least at K=20.

			UpdateHit			
	K	**m**	P	S	F	U
1.	20	1	1574	2724	158	4081
2.	20	2	1598	2767	91	4081
3.	20	4	1610	2799	**47**	4081

Table 3: Training Statistics - Effect of maximum update attempts

In Table 3, we can see that an higher limit on the maximum number of update attempts results in less number of update attempts as expected. A much higher value of m is not preferable because the training updates makes noisy updates in case of *difficult* nodes i.e., the nodes where target transformation is reachable in theory, but is unreachable given the set of features.

			UpdateHit			
	K	**i**	P	S	F	U
1.	1	1	1680	2692	84	4081
2.	1	2	1679	2694	83	4081

Table 4: Training Statistics - Effect of number of iterations

Now, we examine the effect of number of it-

erations on the quality of the model. In table 4, we can observe that the number of iterations on the data has no effect on the quality of the model. This implies, that the model is adequately learnt after one pass through the data. This is possible because of the multiple number of update attempts allowed at every node. Hence, the weights are updated at a node till the model prediction is consistent with the gold transformation.

Based on the above observations, we consider the configuration 4 in Table 2 for the decoding experiments.

Now, we present some of the top features weights leant by the best configuration. The weights convey that important properties of transformation are being learnt well. Table 5 presents the weights of the features 'diceRoot', 'diceRootChildren' and 'diceRootParent'.

Feature	Weight
dice	75.67
diceChildren	540.31
diceParent	595.94
treelet translation probability (ttp) 1	0.77
treelet translation probability (ttp) 2	389.62

Table 5: Weights of dice coefficient based features

We see that the dice coefficient based local and contextual features have a positive impact on the selection of correct transformations. A feature that uses a syntactic language model to compute the perplexity per word has a negative weight of **-1.115**.

Table 6 presents the top-5 entries of contextual features that describe the translation of source argument 'nsubj' using contextual information ('tense' of its parent).

Feature	Weight
roleTenseVib:nsubj+NULL___NULL	44.194196513246
roleTenseVib:nsubj+has_VBN___ne	14.4541356715382
roleTenseVib:nsubj+VBD___ne	10.9241093097953
roleTenseVib:nsubj+VBP___meM	6.14149937079584
roleTenseVib:nsubj+VBP___NULL	4.76795730621754

Table 6: Top weights of a contextual feature : preposition+Tense-postposition

Table 7 presents the top-10 ordering relative position feature where the head word is a verb. In this feature, the relative position (left or right) of the head and the child is captured. For example, a feature 'relPos:amod-NN', if active, conveys that an argument with the role 'amod' is at the left of a head word with POS tag 'NN'.

Feature	Weight
relPos:amod-NN	6.70
relPos:NN-appos	1.62
relPos:lrb-NN	1.62

Table 7: Top weights of *relPos* feature

6.3 Decoding

We computed the translation accuracies using two metrics, (1) BLEU score (Papineni et al., 2002), and (2) Lexical Accuracy (or F-Score) on a test set of 30 sentences. We compared the accuracy of the experimental system (Vaanee) presented in this paper, with Moses (state-of-the-art translation system) and Shakti (rule-based translation system [7]) under similar conditions (with using a development set to tune the models). The rule-based system considered is a general domain system tuned to the tourism domain. The best BLEU score for Moses on the test set is **0.118**, and the best lexical accuracy is **0.512**. The best BLEU score for Shakti is **0.054**, and the best lexical accuracy is **0.369**.

In comparison, the best BLEU score of Vaanee is **0.067**, while the best lexical accuracy is **0.445**. As observed, the decoding results of the experimental system mentioned here are not yet comparable to the state-of-art. The main reasons for the low translation accuracies are,

1. Poor Quality of the dataset

   The dataset currently available for English-Hindi language pair is noisy. This is an extremely large limiting factor for a model which uses rich linguistic information within the statistical framework.

2. Low Parser accuracy

[7] http://shakti.iiit.ac.in/

The parser accuracy on the English-Hindi dataset is low, the reasons being, (1) Noise, (2) Length of sentences, and (3) Wide scope of the tourism domain.

3. Word insertions not implemented yet
4. Non-projectivity not yet handled
5. BLEU is not an appropriate metric

   BLEU is not an appropriate metric (Ananthakrishnan et al.,) for measuring the translation accuracy into Indian languages.

6. Model is context free as far as targets words are concerned. Selection depends on children but not parents and siblings

   This point concerns the decoding algorithm. The current algorithm is greedy while chosing the best translation at every source node. It first explores the K-best local transformations at a source node. It then makes a greedy selection of the predicted subtree based on it's overall score after considering the predictions at the child nodes, and the relative position of the local transformation with respect the predictions at the child nodes.

   The problem in this approach is that, an error once made at a lower level of the tree is propogated to the top, causing more mistakes. A computationally reasonable solution to this problem is to maintain a K-best list of predicted subtrees corresponding to every source node. This allows rectification of a mistake made at any stage.

The system, however, performs better than the rule based system. As observed earlier, the right type of information is being learnt by the model, and the approach looks promising. The limitations expressed here shall be addressed in the future.

7 Conclusion

In this work, we presented a syntax based dependency model to effectively handle problems in translation from English to Indian languages such as, (1) Large word order variation, and (2) Accurate generation of word-forms in the target language by predicted the word and its factors. The model that we have proposed, has the flexibility of adding rich linguistic features.

An experimental version of the system has been implemented, which is available for download at http://shakti.iiit.ac.in/∼sriram/vaanee.html. This can facilitate as a platform for future research in syntax based statistical machine translation from English to Indian languages. We also plan to perform experiments using this system between European languages in future.

The performance of the implemented translation system, is not yet comparable to the state-of-art results primarily for two reasons, (1) Poor quality of available data, because of which our model which uses rich linguistic information doesn't perform as expected, and (2) Components for word insertion and non-projectivity handling are yet to be implemented in this version of the system.

References

Ananthakrishnan, R, B Pushpak, M Sasikumar, and Ritesh Shah. Some issues in automatic evaluation of english-hindi mt: more blues for bleu. ICON-2007.

Ananthakrishnan, R., Jayprasad Hegde, Pushpak Bhattacharyya, and M. Sasikumar. 2008. Simple syntactic and morphological processing can help english-hindi statistical machine translation. In *Proceedings of IJCNLP-2008*. IJCNLP.

Bharati, Akshar, Vineet Chaitanya, Amba P Kulkarni, and Rajeev Sangal. 1997. Anusaaraka: Machine translation in stages. *A Quarterly in Artificial Intelligence, NCST, Bombay (renamed as CDAC, Mumbai)*.

Bharati, Akshar, Medhavi Bhatia, Vineet Chaitanya, and Rajeev Sangal. 1998. Paninian grammar framework applied to english. *South Asian Language Review*, (3).

Bharati, Akshar, Rajeev Sangal, Dipti M Sharma, and Amba P Kulkarni. 2002. Machine translation activities in india: A survey. In *Proceedings of workshop on survey on Research and Development of Machine Translation in Asian Countries*.

Chiang, David. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 263–270, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Crammer, K., R. McDonald, and F. Pereira. 2005. Scalable large-margin online learning for structured classification. Technical report, University of Pennsylvania.

Gali, Karthik and Sriram Venkatapathy. 2009. Sentence realisation from bag of words with dependency constraints. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium*, pages 19–24, Boulder, Colorado, June. Association for Computational Linguistics.

Guo, Yuqing, Josef van Genabith, and Haifeng Wang. 2008. Dependency-based n-gram models for general purpose sentence realisation. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 297–304, Manchester, UK, August. Coling 2008 Organizing Committee.

Ittycheriah, Abraham and Salim Roukos. 2007. Direct translation model 2. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 57–64, Rochester, New York, April. Association for Computational Linguistics.

Klein, Dan and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan, July. Association for Computational Linguistics.

Koehn, Philipp and Hieu Hoang. 2007. Factored translation models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876.

Koehn, P., F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference 2003 (HLT-NAACL 2003)*, Edmonton, Canada, May.

Liang, P., A. Bouchard-Cote, D. Klein, and B. Taskar. 2006. An end-to-end discriminative approach to machine translation. In *International Conference on Computational Linguistics and Association for Computational Linguistics (COLING/ACL)*.

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

Och, F.J. and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Papineni, Kishore, Salim Roukos, Todd Ward, and W.J. Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of 40^{th} Annual Meeting of the Association of Computational Linguistics*, pages 313–318, Philadelphia, PA, July.

Quirk, Chris, Arul Menezes, and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Ramanathan, Ananthakrishnan, Hansraj Choudhary, Avishek Ghosh, and Pushpak Bhattacharyya. 2009. Case markers and morphology: Addressing the crux of the fluency problem in english-hindi smt. In *Proceedings of ACL-IJCNLP 2009*. ACL-IJCNLP.

Shen, Libin, Jinxi Xu, Bing Zhang, Spyros Matsoukas, and Ralph Weischedel. 2009. Effective use of linguistic and contextual information for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 72–80, Singapore, August. Association for Computational Linguistics.

Vaidya, Ashwini, Samar Husain, Prashanth Reddy, and Dipti M Sharma. 2009. A karaka based annotation scheme for english. In *Proceedings of CICLing , 2009*.

Wu, Dekai. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377–404.

Yamada, Kenji and Kevin Knight. 2002. A decoder for syntax-based statistical mt. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 303–310, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

Improved Language Modeling for English-Persian Statistical Machine Translation

Mahsa Mohaghegh
Massey University,
School of Engineering and Advanced Technology
m.mohaghegh@massey.ac.nz

Abdolhossein Sarrafzadeh
Unitec,
Department of computing
hsarrafzadeh@unitec.ac.nz

Tom Moir
Massey University,
School of Engineering and Advanced Technology
T.J.Moir@massey.ac.nz

Abstract

As interaction between speakers of different languages continues to increase, the ever-present problem of language barriers must be overcome. For the same reason, automatic language translation (Machine Translation) has become an attractive area of research and development. Statistical Machine Translation (SMT) has been used for translation between many language pairs, the results of which have shown considerable success. The focus of this research is on the English/Persian language pair. This paper investigates the development and evaluation of the performance of a statistical machine translation system by building a baseline system using subtitles from Persian films. We present an overview of previous related work in English/Persian machine translation, and examine the available corpora for this language pair. We finally show the results of the experiments of our system using an in-house corpus and compare the results we obtained when building a language model with different sized monolingual corpora. Different automatic evaluation metrics like BLEU, NIST and IBM-BLEU were used to evaluate the performance of the system on half of the corpus built. Finally, we look at future work by outlining ways of getting highly accurate translations as fast as possible.

1 Introduction

Over the 20th century, international interaction, travel and business relationships have increased enormously. With the entrance of the World Wide Web effectively connecting countries together over a giant network, this interaction reached a new peak. In the area of business and commerce, the vast majority of companies simply would not work without this global connection. However, with this vast global benefit comes a global problem: the language barrier. As the *international* connection barriers continually break down, the *language* barrier becomes a greater issue. The English language is now the world's lingua franca, and non-English speaking people are faced with the problem of communication, and limited access to resources in English.

Machine translation is the process of using computers for translation from one human language to another(Lopez, 2008). This is not a recent area of research and development. In fact, machine translation was one of the first applications of natural language processing, with research work dating back to the 1950s(Cancedda, Dymetman, Foster, & Goutte, 2009). However, due to the complexity and diversity of human language, automated translation is one of the hardest problems in computer science, and significantly successful results are uncommon.

There are a number of different approaches to machine translation. Statistical Machine Translation (SMT) however, seems to be the preferred approach of many industrial and academic research laboratories (Schmidt, 2007). The advantages of SMT compared to rule-based approaches lie in their adaptability to different domains and languages: once a functional

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 75–82,
COLING 2010, Beijing, August 2010.

system exists, all that has to be done in order to make it work with other language pairs or text domains is to train it on new data.

Research work on statistical machine translation systems began in the early 1990s. These systems, which are based on phrase-based approaches, operate using parallel corpora – huge databases of corresponding sentences in two languages, and employ statistics and probability to learn by example which translation of a word or phrase is most likely correct. The translation moves directly from source language to target language with no intermediate transfer step. In recent years, such phrase-based MT approaches have become popular because they generally show better translation results. One major factor for this development is the growing availability of large monolingual and bilingual text corpora in recent years for a number of languages.

The focus of this paper is on statistical machine translation for the English/Persian language pair. The statistical approach has only been employed in several experimental translation attempts for this language pair, and is still largely undeveloped. This project is considered to be a challenge for several reasons. Firstly, the Persian language structure is very different in comparison to English; secondly, there has been little previous work done for this language pair; and thirdly, effective SMT systems rely on very large bilingual corpora, however these are not readily available for the English/Persian language pair.

1.1 The Persian Language

The Persian language, or Farsi as it is also known as, belongs to the Indo-European language family and is one of the more dominant languages in parts of the Middle East. It is in fact the most widely spoken language in the Iranian branch of the Indo-Iranian languages, being the official language of Iran (Persia) and also spoken in several countries including Iran, Tajikistan and Afghanistan. There also exist large groups and communities in Iraq, United Arab Emirates, People's Democratic Republic of Yemen, Bahrain, and Oman, not to mention communities in the USA.

Persian uses a script that is written from right to left. It has similarities with Arabic but has an extended alphabet and different words and/or pronunciations from Arabic.

During its long history, the language has been influenced by other languages such as Arabic, Turkish and even European languages such as English and French. Today's Persian contains many words from these languages and in some cases words from other languages still follow the grammar of their original language particularly in building plural, singular or different verb forms. Because of the special and different nature of the Persian language compared to other languages like English, the design of SMT systems for Persian requires special considerations.

1.2 Related Work

Several MT systems have already been constructed for the English/Persian language pair.

One such system is the Shiraz project, (Amtrup, Laboratory, & University, 2000). The Shiraz MT system is an MT prototype that translates text one way from Persian to English. The project began in 1997 and the final version was delivered in 1999.

The Shiraz corpus is a 10 MB manually-constructed bilingually tagged Persian to English dictionary of about 50,000 words, developed using on-line material for testing purposes in a project at New Mexico State University. The system also comprises its own syntactic parser and morphological analyzer, and is focused on news stories material translation as its domain.

Another English/Persian system was developed by (Saedi, Motazadi, & Shamsfard, 2009). This system, called PEnTrans, is a bidirectional text translator, comprising two main modules (PEnT1, and PEnT2) which translate in opposite directions (PEnT1 from English to Persian; PEnT2 from Persian to English). PEnT1 employs a combination of both corpus based and extended dictionary approaches, and PEnT2 uses a combination of rule, knowledge and corpus based approaches. PEnTrans introduced a new WSD method with a hybrid measure which evaluates different word senses in a

sentence and scores them according to their condition in the sentence, together with the placement of other words in that sentence.

ParsTranslator is a machine translation system built to translate English to Persian text. It was first released for public use in mid-1997, the latest update being PTran version in April 2004. The ParsTran input uses English text typed or from a file. The latest version is able to operate for over 1.5 million words and terminologies in English. It covers 33 fields of sciences, and is a growing translation service, with word banks being continually reviewed and updated, available at: http://www.ParsTranslator.Net/eng/index.htm. Another English to Persian MT system is the rule-based system developed by (Faili & Ghassem-Sani, 2005)This system was based on tree adjoining grammar (TAG), and later improved by implementing trained decision trees as a word sense disambiguation module.

Mohaghegh et al. (2009) presented the first such attempt to construct a parallel corpus from BBC news stories. This corpus is intended to be an open corpus in which more text may be added as they are collected. This corpus was used to construct a prototype for the first statistical machine translation system. The problems encountered, especially with the process of alignment are discussed in this research (Mohaghegh & Sarrafzadeh, 2009).

Most of these systems have largely used a rule based approach, and their BLEU scores on a standard data set have not been published. Nowadays however, most large companies employ the statistical translation approach, using exceedingly large amounts of bilingual data (aligned sentences in two languages). A good example of this is perhaps the most well-known Persian/English MT system: Google Translate recently released option for this language pair. Google's MT system is based on the statistical approach, and was made available online as a BETA version in June 2009.

The Transonics Spoken Dialogue Translator is also partially a statistically based machine translation system. The complete system itself operates using a speech to text converter, statistical language translation, and subsequent text to speech conversion. The actual translation unit operates in two modes: in-domain and out-of-domain. A classifier attempts to assign a concept to an utterance. If the object to be translated is within the translation domain, the system is capable of significantly accurate translations. Where the object is outside the translation domain, the SMT method is used. Transonics is a translation system for a specific domain (medical: doctor-to-patient interviews), and only deals with question/answer situations (Ettelaie, et al., 2005).

Another speech-to-speech English/Persian machine translation system is suggested by Xiang et al. They present an unsupervised training technique to alleviate the problem of the lack of bilingual training data by taking advantage of available source language data(Xiang, Deng, & Gao, 2008).

However, there was no large parallel text corpus available at the time of development for both of these systems. For its specific domain, the Transonics translation system relied on a dictionary approach for translation, using a speech corpus, rather than a parallel text corpus. Their Statistical Translation approach was merely used as a backup system.

2 Corpus Development for Persian

A corpus is defined as a large compilation of written text or audible speech transcript. Corpora, both monolingual and bilingual, have been used in various applications in computational linguistics and machine translation.

A parallel corpus is effectively two corpora in two different languages comprising sentences and phrases accurately translated and aligned together phrase to phrase. When used in machine translation systems, parallel corpora must be of a very large size – billions of sentences – to be effective. It is for this reason that the Persian language poses some difficulty. There is an acute shortage of digitally stored linguistic material, and few parallel online documents, making the construction of a parallel Persian corpus is extremely difficult.

There are a few parallel Persian corpora that do exist. These vary in size, and in the domains they cover. One such corpus is FLDB1, which is a linguistic corpus consisting of approximately 3 million words in ASCII format. This corpus

was developed and released by (Assi, 1997) at the Institute for Humanities and Cultural Studies. This corpus version was updated in 2005, in 1256 character code page, and named PLDB2. This new updated version contains more than 56 million words, and was constructed with contemporary literary books, articles, magazines, newspapers, laws and regulations, transcriptions of news, reports, and telephone speeches for lexicography purposes.

Several corpora construction efforts have been made based on online Hamshahri newspaper archives. These include Ghayoomi (2004), with 6 months of Hamshahri archives to yield a corpus of 6.5 million words, and (Darrudi, Hejazi, & Oroumchian, 2004), with 4 years' worth of archives to yield a 37 million-word corpus.

The 'Peykareh' or 'Text Corpus' is a corpus of 38 million words developed by Bijankhan et al. available at: http://ece.ut.ac.ir/dbrg/bijankhan/ and comprises newspapers, books, magazines articles, technical books, together with transcription of dialogs, monologues, and speeches for language modeling purposes.

Shiraz corpus (Amtrup, et al., 2000)is a bilingual tagged corpus of about 3000 aligned Persian/English sentences also collected from the Hamshahri newspaper online archive and manually translated at New Mexico State University.

Another corpus, TEP (Tehran English-Persian corpus), available at: http://ece.ut.ac.ir/NLP/_resources.htm , consists of 21,000 subtitle files obtained from www.opensubtitles.org. Subtitle pairs of multiple versions of same movie were extracted, a total of about 1,200(Itamar & Itai, 2008) then aligned the files using their proposed dynamic programming method. This method operates by using the timing information contained in subtitle files so as to align the text accurately. The end product yielded a parallel corpus of approximately 150,000 sentences which has 4,100,000 tokens in Persian and 4,400,000 tokens in English.

Finally, European Language Resources Association (ELRA), available at: http://catalog.elra.info/product_info.php?products_id=1111, have constructed a corpus which consists of about 3,500,000 English and Persian words aligned at sentence level, to give approximately 100,000 sentences distributed over 50,021 entries. The corpus was originally constructed with SQL Server, but presented in access type file. The format for the files is Unicode. This corpus consists of several different domains, including art, culture, idioms, law, literature, medicine, poetry, politics, proverbs, religion, and science; it is available for sale online.

3 Statistical Machine Translation

3.1 General

Statistical machine translation (SMT) can be defined as the process of maximizing the probability of a sentence s in the source language matching a sentence t in the target language. In other words, *"given a sentence s in the source language, we seek the sentence t in the target language such that it maximizes P(t | s) which is called the conditional probability or the chance of t happening given s"* (Koehn, et al., 2007).

It is also referred to as the most likely translation. This can be more formally written as shown in equation (1).

$$arg\ max\ P(t \mid s) \qquad (1)$$

Using Bayes Rule from equation (2), we can write equation (1) for the most likely translation as shown in equation (3).

$$P(t \mid s) = P(t) * P(s \mid t) = P(s) \qquad (2)$$

$$\arg\max P(t \mid s) = \arg\max P(t) * P(s \mid t) \qquad (3)$$

Where (t) is the target sentence, and (s) is the source sentence. P (t) is the target language model and P(s | t) is the translation model. The argmax operation is the search, which is done by a so-called decoder which is a part of a statistical machine translation system.

3.2 Statistical Machine Translation Tools

There are a number of implementations of subtasks and algorithms in SMT and even software tools that can be used to set up a fully-featured state-of-the-art SMT system.

Moses (Koehn, et al., 2007) is an open-source statistical machine translation system which allows one to train translation models using GIZA++ (Och & Ney, 2004).for any given language pair for which a parallel corpus exists. This tool was used to build the baseline system discussed in this paper. MOSES uses a beam search algorithm where the translated output sentence is generated left to right in form of hypotheses. Beam-search is an efficient search algorithm which quickly finds the highest probability translation among the exponential number of choices.

The search begins with an initial state where no foreign input words are translated and no English output words have been generated. New states are created by extending the English output with a phrasal translation of that covers some of the foreign input words not yet translated.

The algorithm can be used for exhaustively searching through all possible translations when data gets very large. The search can be optimized by discarding hypotheses that cannot be part of the path to the best translation. Furthermore, by comparing states, one can define a beam of good hypotheses and prune out hypotheses that fall out of this beam (Dean & Ghemawat, 2008).

3.3 Building a Baseline SMT System

To build a good baseline system it is important to build a sentence aligned parallel corpus which is spell-checked and grammatically correct for both the source and target language. The alignment of words or phrases turns out to be the most difficult problem SMT faces.

Words and phrases in the source and target languages normally differ in where they are placed in a sentence. Words that appear on one language side may be dropped on the other. One English word may have as its counterpart a longer Persian phrase and vice versa. The accuracy of SMT relies heavily on the existence of large amounts of data which is commonly referred to as a parallel corpus. The first step taken was to develop the parallel corpus. This corpus is intended to be an open corpus in which more text can be added as they are collected. Sentences were aligned using Microsoft's bi-lingual sentence aligner developed by (Moore, 2002).

The next step we plan to take involves the construction of a statistical prototype based on the largest available English/Persian parallel corpus extracted from the domain of movie subtitles. This domain was chosen because the maximum number of words that can be displayed as a subtitle on the screen is between 10- 12 which means both training and decoding will be a lot faster. Building a parallel corpus for any domain is generally the most time consuming process as it depends on the availability of parallel text. But the domain of subtitling makes it easier to get the source language in the form of scripts and the target language in the form of subtitles in many different languages.

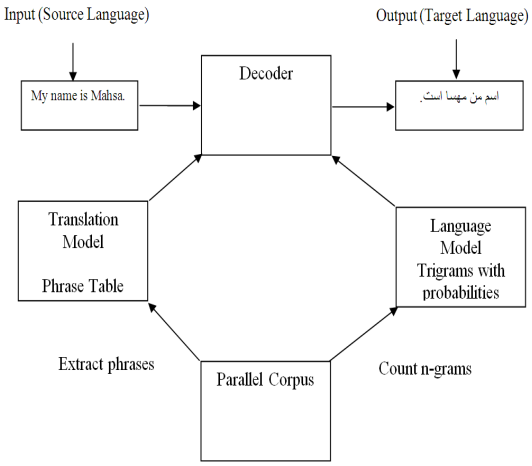


Figure1. A typical SMT System

A language model (LM) is usually trained on large amounts of monolingual data in the target language to ensure the fluency of the language that the sentence is getting translated into. Language modeling is not only used in machine translation but also used in many natural language processing applications such as speech recognition, part-of-speech tagging, parsing and information retrieval. A statistical language model assigns probabilities to a sequence of words and tries to capture the properties of a language.

The Language Model (LM) for this study was trained on the BBC Persian News corpus and also an in-house corpus from different genres. The SRILM toolkit developed was used

to train a 5-gram LM for experimentation as in (Stolcke, 2002).

4 Experiments and Results

4.1 Experiment setup

We used Moses a phrase-based SMT development tool for constructing our machine translation system. This included n-gram language models trained with the SRI language modeling tool, GIZA++ alignment tool, Moses decoder and the script to induce phrase-based translation models from word-based ones.

4.2 Performance evaluation metrics

A lot of research has been done in the field of automatic machine translation evaluation. Human evaluations of machine translation are extensive but expensive. Human evaluations can take months to finish and involve human labor that cannot be reused which is the main idea behind the method of automatic machine translation evaluation that is quick, inexpensive, and language independent.

One of the most popular metrics is called BLEU (BiLingual Evaluation Understudy) developed at IBM. The closer a MT is to a professional human translation, the better it is. This is the central idea behind the BLEU metric. NIST is another automatic evaluation metric with the following primary differences compared to BLEU such as Text pre-processing, gentler length penalty, information-weighted N-gram counts and selective use of N-grams (Li, Callison-Burch, Khudanpur, & Thornton, 2009); (Li, Callison-Burch, Khudanpur, & Thornton, 2009).

4.3 Discussion and analysis of the results

In this study, Moses was used to establish a baseline system. This system was trained and tested on three in-house corpora, the first 817 sentences, the second 1011 sentences, and the third 2343 sentences. The data available was split into a training and test set. Microsoft's bilingual sentence aligner (Moore, 2002) was used to align the corpus and training sets. Aligning was also performed manually to aid in the improvement of the results. As the corpus size increased, we performed various experiments such as increasing the language model in each instance.

Test No.	EN/FA 1	EN/FA 2	EN/FA 3
Test Sentences	817	1011	2343
Training Sentences	864	1066	7005

Table 1. Size of test set and train set (language Model) En: English, FA: Farsi

Evaluation results from these experiments are presented in Tables 2, 3 and 4. As expected, BLEU scores improved as the size of the corpus increased. The BLEU scores themselves were significantly low; however this was expected due to the small size of the corpus. We plan to update and increase the corpus size in the near future, which will undoubtedly yield more satisfactory results.

LM=864	BLEU	NIST	IBM-BLEU
Corpus size 817	0.1061	1.8218	0.0060
Corpus size 1011	0.0882	1.5338	0.0050
Corpus size 2343	0.0806	1.7364	0.0067

Table 2. Result obtained using Language Model size=864

LM=1066	BLEU	NIST	IBM-BLEU
Corpus size 817	0.0920	1.6838	0.0060
Corpus size 1011	0.0986	1.5301	0.0050
Corpus size 2343	0.1127	1.6961	0.0069

Table 3. Result obtained using Language Model size=1066

LM= 7005	BLEU	NIST	IBM-BLEU
Corpus size 817	0.0805	1.6721	0.0063
Corpus size 1011	0.0888	1.5512	0.0051
Corpus size 2343	0.1148	1.7554	0.0071

Table 4. Result obtained using Language Model size=7005

The first test was performed on a corpus of 817 sentences in Persian and the same number for their aligned translation in English. In this instance, the training set used was 864 sentences. Results of this translation were evaluated using three evaluation metrics (BLEU, NIST, and IBM-BLEU) An excerpt from the output of this first experiment is shown in figure2 (a).

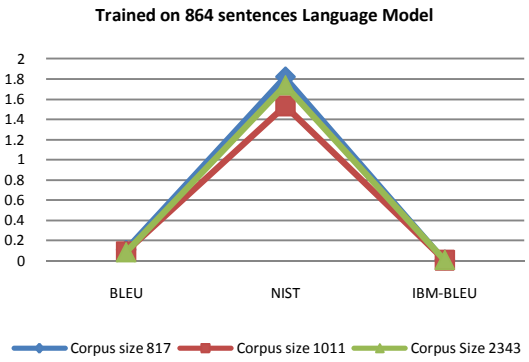
The second test comprised of a 1011 sentences corpus, with a 1066 sentence training set. As can be seen, the evaluation metric results improved.

The same experiment was repeated for a third time, this time with an even larger corpus of 2343 sentences, and a training set of 7005 sentences. The result can be seen in table 4. The results obtained in this test were close to those in the previous test, apart from a small increase in BLEU scores. It must be noted that BLEU is only a tool to compare different MT systems. So an increase in BLEU scores may not necessarily mean an increase in the accuracy of translation. The performance of the baseline English-Persian SMT system was evaluated by computing BLEU, IBM-BLEU-NIST (Li, et al., 2009) scores from different automatic evaluation metrics against different sizes of the sentence aligned corpus and different sizes of the training set .

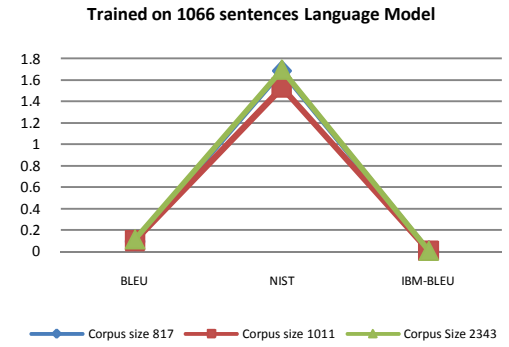
Tables 2, 3 and 4 show the results obtained using corpuses of 817, 1011, and 2343 sentences respectively. The language model size was varied from 864 to 1066 and finally to 7005 sentences.

Moreover as shown in table 3, using a corpus and language model of 1011 and 1066 in size respectively produces better results. This can clearly be noticed from graph in Figure 2(b).

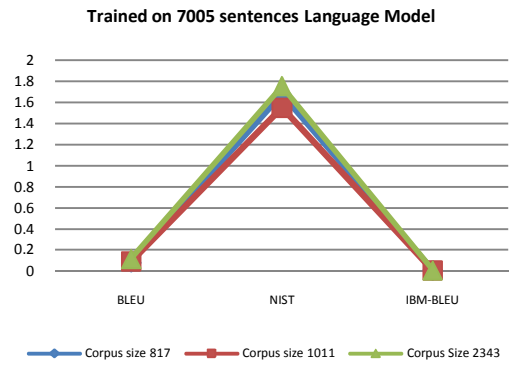
Finally, increasing the size of the corpus to 2343 and language model constructed using 7005 sentences produced the best translation results as shown in both Figure 2(c) and Table 4. This data shows that an increased corpus size will yield an improved translation quality, but only as long as the size of the language model is proportional to the corpus size. Literature refers to the fact that the size of the corpus, although important, does not have as great an effect as corpus and language model in the domain of translation (Ma & Way, 2009). In the Persian language, some problems and difficulties arise due to natural language ambiguities, anaphora resolution, idioms and differences in the types and symbols used for punctuation. These issues had to be resolved before any attempt at SMT could be made. Needless to stress on the fact that the better the alignment the better the results of the translation.



(a)



(b)



(c)

Figure 3. (a) Results obtained using training size=864 (b) Results obtained using training size=1066 (c) Results obtained using training size=7005

5 Future work

Despite the fact that compared to other language pairs, the available parallel corpora for the English/Persian language pair is significantly smaller, the future of statistical machine translation for this language pair looks promising. We have been able to procure several very large bilingual corpora, which we intend to combine with the open corpus we used in the original tests. With the use of a much larger bilingual corpus, we expect to produce a significantly higher evaluation metric score. Our planned immediate future work will consist of combining these corpora together, addressing the task of corpus alignment, and continuing the use of a web crawler to obtain further bilingual text.

6 Conclusion

This paper presented an overview of some of the work in the area of English/Persian MT systems that has been done to date, and showed a set of experiments in which our SMT system was applied to the Persian language using a relatively small corpus. The first part of this work was to test how well our system translates from Persian to English when trained on the available corpora and to spot and try and resolve problems with the process and the output produced. According to the results we obtained, it was concluded that a corpus of much greater size would be required to produce satisfactory results. Our experience with the corpus of smaller size shows us that for a large corpus, there will be a significant amount of work required in aligning sentences.

References

Amtrup, J., Laboratory, C. R., & University, N. M. S. (2000). *Persian-English machine translation: An overview of the Shiraz project*: Computing Research Laboratory, New Mexico State University.

Assi, S. (1997). Farsi linguistic database (FLDB). *International Journal of Lexicography, 10*(3), 5.

Cancedda, N., Dymetman, M., Foster, G., & Goutte, C. (2009). A Statistical Machine Translation Primer.

Darrudi, E., Hejazi, M., & Oroumchian, F. (2004). *Assessment of a modern farsi corpus*.

Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

Ettelaie, E., Gandhe, S., Georgiou, P., Knight, K., Marcu, D., Narayanan, S., et al. (2005). *Transonics: A practical speech-to-speech translator for English-Farsi medical dialogues*.

Faili, H., & Ghassem-Sani, G. (2005). *Using Decision Tree Approach For Ambiguity Resolution In Machine Translation*.

Itamar, E., & Itai, A. (2008). *Using Movie Subtitles for Creating a Large-Scale Bilingual Corpora*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., et al. (2007). *Moses: Open source toolkit for statistical machine translation*.

Li, Z., Callison-Burch, C., Khudanpur, S., & Thornton, W. (2009). Decoding in Joshua. *The Prague Bulletin of Mathematical Linguistics, 91*, 47-56.

Lopez, A. (2008). Statistical machine translation.

Ma, Y., & Way, A. (2009). *Bilingually Motivated Domain-Adapted Word Segmentation for Statistical Machine Translation*.

Mohaghegh, M., & Sarrafzadeh, A. (2009). *An analysis of the effect of training data variation in English-Persian statistical machine translation*.

Moore, R. (2002). Fast and accurate sentence alignment of bilingual corpora. *Lecture notes in computer science*, 135-144.

Och, F., & Ney, H. (2004). The alignment template approach to statistical machine translation. *Computational Linguistics, 30*(4), 417-449.

Saedi, C., Motazadi, Y., & Shamsfard, M. (2009). *Automatic translation between English and Persian texts*.

Schmidt, A. (2007). *Statistical Machine Translation Between New Language Pairs Using Multiple Intermediaries*.

Stolcke, A. (2002). *SRILM-an extensible language modeling toolkit*.

Xiang, B., Deng, Y., & Gao, Y. (2008). *Unsupervised training for farsi-english speech-to-speech translation*.

Manipuri-English Bidirectional Statistical Machine Translation Systems using Morphology and Dependency Relations

Thoudam Doren Singh
Department of Computer Science and Engineering
Jadavpur University
thoudam.doren@gmail.com

Sivaji Bandyopadhyay
Department of Computer Science and Engineering
Jadavpur University
sivaji_cse_ju@yahoo.com

Abstract

The present work reports the development of Manipuri-English bidirectional statistical machine translation systems. In the English-Manipuri statistical machine translation system, the role of the suffixes and dependency relations on the source side and case markers on the target side are identified as important translation factors. A parallel corpus of 10350 sentences from news domain is used for training and the system is tested with 500 sentences. Using the proposed translation factors, the output of the translation quality is improved as indicated by baseline BLEU score of 13.045 and factored BLEU score of 16.873 respectively. Similarly, for the Manipuri English system, the role of case markers and POS tags information at the source side and suffixes and dependency relations at the target side are identified as useful translation factors. The case markers and suffixes are not only responsible to determine the word classes but also to determine the dependency relations. Using these translation factors, the output of the translation quality is improved as indicated by baseline BLEU score of 13.452 and factored BLEU score of 17.573 respectively. Further, the subjective evaluation indicates the improvement in the fluency and adequacy of both the factored SMT outputs over the respective baseline systems.

1 Introduction

Manipuri has little resource for NLP related research and development activities. Manipuri is a less privileged Tibeto-Burman language spoken by approximately three million people mainly in the state of Manipur in India as well as its neighboring states and in the countries of Myanmar and Bangladesh. Some of the unique features of this language are tone, the agglutinative verb morphology and predominance of aspect than tense, lack of grammatical gender, number and person. Other features are verb final word order in a sentence i.e., Subject Object Verb (SOV) order, extensive suffix with more limited prefixation. In Manipuri, identification of most of the word classes and sentence types are based on the markers. All sentences, except interrogatives end with one of these mood markers, which may or may not be followed by an enclitic. Basic sentence types in Manipuri are determined through illocutionary mood markers, all of which are verbal inflectional suffixes, with the exception of the interrogatives that end with an enclitic. Two important problems in applying statistical machine translation (SMT) techniques to English-Manipuri bidirectional MT systems are: (a) the wide syntactic divergence between the language pairs, and (b) the richer morphology and case marking of Manipuri compared to English. The first problem manifests itself in poor word-order in the output translations, while the second one leads to incorrect inflections and case marking. The output Manipuri sentences in case of English-Manipuri system suffer badly when morphology and case markers are incorrect in this free word order and morphologically rich language.

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 83–91,
COLING 2010, Beijing, August 2010.

The parallel corpora used is in news domain which have been collected, cleaned and aligned (Singh et al., 2010b) from the Sangai Express newspaper website www.thesangaiexpress.com available in both Manipuri and English. A daily basis collection was done covering the period from May 2008 to November 2008 since there is no repository.

2 Related Works

Koehn and Hoang (2007) developed a framework for statistical translation models that tightly integrates additional morphological, syntactic, or semantic information. Statistical Machine Translation with scarce resources using morpho-syntactic information is discussed in (Nieβen and Ney, 2004). It introduces sentence level restructuring transformations that aim at the assimilation of word order in related sentences and exploitation of the bilingual training data by explicitly taking into account the interdependencies of related inflected forms thereby improving the translation quality. Popovic and Ney (2006) discussed SMT with a small amount of bilingual training data. Case markers and morphology are used to address the crux of fluency in the English-Hindi SMT system (Ramanathan et al., 2009). Work on translating from rich to poor morphology using factored model is reported in (Avramidis and Koehn, 2008). In this method of enriching input, the case agreement for nouns, adjectives and articles are mainly defined by the syntactic role of each phrase. Resolution of verb conjugation is done by identifying the person of a verb and using the linguistic information tag. Manipuri to English Example Based Machine Translation system is reported in (Singh and Bandyopadhyay, 2010a) on news domain. For this, POS tagging, morphological analysis, NER and chunking are applied on the parallel corpus for phrase level alignment. Chunks are aligned using a dynamic programming "edit-distance style" alignment algorithm. The translation process initially looks for an exact match in the parallel example base and returns the retrieved target output. Otherwise, the maximal match source sentence is identified. For word level mismatch, the unmatched words in the input are either translated from the lexicon or transliterated. Unmatched phrases are looked into the phrase level parallel example base; the target phrase translations are identified and then recombined with the retrieved output. English-Manipuri SMT system using morpho-syntactic and semantic information is reported in (Singh and Bandyopadhyay, 2010c). In this system, the role of the suffixes and dependency relations on the source side and case markers on the target side are identified as important translation factors.

3 Syntactic Reordering

This is a preprocessing step applied to the input English sentences for English-Manipuri SMT system. The program for syntactic reordering uses the parse trees generated by Stanford parser[1] and applies a handful of reordering rules written using perl module Parse::RecDescent. By doing this, the SVO order of English is changed to SOV order for Manipuri, and post modifiers are converted to pre-modifiers. The basic difference of Manipuri phrase order compared to English is handled by reordering the input sentence following the rule (Rao et al., 2000):

$SS_mV\ V_mOO_mC_m \rightarrow C'_mS'_mS'O'_mO'V'_mV'$

where, S: Subject
O: Object
V : Verb
C_m: Clause modifier
X': Corresponding constituent in Manipuri, where X is S, O, or V
X_m: modifier of X

There are two reasons why the syntactic reordering approach improves over the baseline phrase-based SMT system (Wang et al., 2007). One obvious benefit is that the word order of the transformed source sentence is much closer to the target sentence, which reduces the reliance on the distortion model to perform reordering during decoding. Another potential benefit is that the alignment between the two sides will be of higher quality because of fewer "distortions" between the source and the target, so that the resulting phrase table of the reordered system would be better. However, a counter argument is that the reordering is very error prone, so that the added noise in the reordered data actually hurts the alignments and hence the phrase tables.

[1] http://nlp.stanford.edu/software/lex-parser.shtml

4 Morphology

The affixes are the determining factor of the word class in Manipuri. In this agglutinative language the number of verbal suffixes is more than that of nominal suffixes. Works on Manipuri morphology are found in (Singh and Bandyopadhyay, 2006) and (Singh and Bandyopadhyay, 2008). In this language, a verb must minimally consist of a verb root and an inflectional suffix. A noun may be optionally affixed by derivational morphemes indicating gender, number and quantity. Further, a noun may be prefixed by a pronominal prefix which indicates its possessor. Words in Manipuri consist of stems or bound roots with suffixes (from one to ten suffixes), prefixes (only one per word) and/or enclitics.

(a) ইবোমচা-না বোল-দু কাওই
Ibomcha-na *Ball-du* *kao-i*
Ibomcha-nom Ball-distal kick
Ibomcha kicks the ball.

(b) বোল-দু ইবোমচা-না কাওই
Ball-du *Ibomcha-na* *kao-i*
Ball-distal Ibomcha-nom kick
Ibomcha kicks the ball.

The identification of subject and object in both the sentences are done by the suffixes না (*na*) and দু (*du*) as given by the examples (a) and (b). The case markers convey the right meaning during translation though the most acceptable order of Manipuri sentence is SOV. In order to produce a good translation output all the morphological forms of a word and its translations should be available in the training data and every word has to appear with every possible suffixes. This will require a large training data. By learning the general rules of morphology, the amount of training data could be reduced. Separating lemma and suffix allows the system to learn more about the different possible word formations.

Manipuri	Gloss	English Meaning
তোম্না	*Tom-na*	by Tom
তোমদগী	*Tom-dagi*	from Tom
তোমসু	*Tom-su*	Tom also
তোমগী	*Tom-gi*	of Tom
তোমগা	*Tom-ga*	with Tom

Table 1: Some of the inflected forms of names in Manipuri and its corresponding English meaning

Table 1 gives some examples of the inflected forms of a person name and its corresponding English meaning. The Manipuri stemmer separates the case markers such as –না *(-na)*, -দগী *(-dagi)*, -সু *(-su)*, -গী (*-gi*), -গা *(-ga)* etc. from surface forms so that "তোম" (*Tom*) from Manipuri side matches with "Tom" at English side helping to overcome the data sparseness. Enclitics in Manipuri fall into six categories: determiners, case markers, the copula, mood markers, inclusive / exclusive and pragmatic peak markers and attitude markers. The role of the enclitics used and its meaning differs based on the context.

5 Factored Model of Translation

Using factored approach, a tighter integration of linguistic information into the translation model is done for two reasons[2]:

- Translation models that operate on more general representations, such as lemma instead of surface forms of words, can draw on richer statistics and overcome the data sparseness problem caused by limited training data.
- Many aspects of translation can be best explained at a morphological, syntactic or semantic level. Having such information available to the translation model allows the direct modeling of these aspects. For instance, reordering at the sentence level is mostly driven by general syntactic principles, local agreement constraints that show up in morphology, etc.

5.1 Combination of Components in Factored Model

Factored translation model is the combination of several components including language model, reordering model, translation steps and generation steps in a log-linear model[3]:

$$p(\mathbf{e}|\mathbf{f}) = \frac{1}{Z}\sum_{i=1}^{n}\lambda_i h_i(\mathbf{e},\mathbf{f}) \quad (1)$$

Z is a normalization constant that is ignored in practice. To compute the probability of a translation **e** given an input sentence **f**, we have to evaluate each feature function h_i. The feature weight

[2]http://www.statmt.org/moses/?n=Moses.FactoredModels
[3]http://www.statmt.org/moses/?n=Moses.FactoredModels

λ_i in the log linear model is determined by using minimum error rate training method (Och, 2003).

For a translation step component, each feature function h_t is defined over the phrase pairs (f_j,e_j) given a scoring function τ:

$$h_t(\mathbf{e},\mathbf{f}) = \sum_j \tau(f_j, e_j) \quad (2)$$

For the generation step component, each feature function h_g given a scoring function γ is defined over the output words e_k only:

$$h_g(\mathbf{e},\mathbf{f}) = \sum_k \gamma(e_k) \quad (3)$$

5.2 Stanford Dependency Parser

The dependency relations used in the experiment are generated by the Stanford dependency parser (Marie-Catherine de Marneffe and Manning, 2008). This parser uses 55 relations to express the dependencies among the various words in a sentence. The dependencies are all binary relations: a grammatical relation holds between a governor and a dependent. These relations form a hierarchical structure with the most general relation at the root.

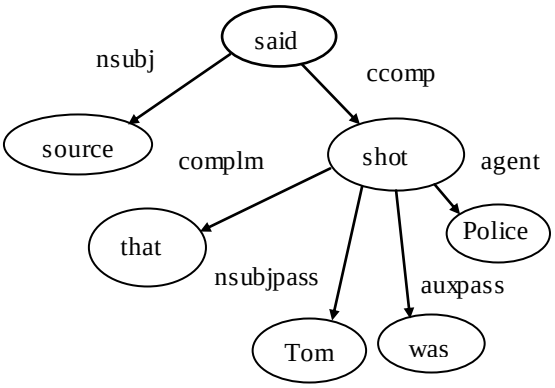


Figure 1. Dependency relation graph of the sentence "*Sources said that Tom was shot by police*" generated by Stanford Parser

There are various argument relations like subject, object, objects of prepositions and clausal complements, modifier relations like adjectival, adverbial, participial, infinitival modifiers and other relations like coordination, conjunct, expletive and punctuation. Let us consider an example "*Sources said that Tom was shot by police*". Stanford parser produces the dependency relations, nsubj(said, sources) and agent (shot, police) . Thus, *sources*|nsubj and *police*|agent are the factors used. "Tom was shot by police" forms the object of the verb "*said*". The Stanford parser represents these dependencies with the help of a clausal complement relation which links "*said*" with "*shot*" and uses the complementizer relation to introduce the subordination conjunction. Figure 1 shows the dependency relation graph of the sentence "*Sources said that Tom was shot by police*".

5.3 Factorization approach of English-Manipuri SMT system

Manipuri case markers are decided by dependency relation and aspect information of English. Figure 2 shows the translation factors used in the translation between English and Manipuri.

(i) Tomba drives the car.

তোম্বনা কারদু খৌই

Tomba-na car-du thou-i

(Tomba) (the car) (drives)

Tomba|empty|nsubj drive|s|empty the|empty|det car|empty|dobj

A subject requires a case marker in a clause with a perfective form such as –না *(na)*. It can be represented as,

suffix+ dependency relation → case marker

s|empty + empty|dobj → না (*na*)

(ii) Birds are flying.

উচেকশিং পাইরি

ucheksing payri

(birds are) (flying)

Bird|s|nsubj are|empty|aux fly|ing|empty

Thus, English-Manipuri factorization consists of

- a lemma to lemma translation factor [i.e., Bird → উচেক *(uchek)*]
- a suffix + dependency relation → suffix [i.e., s + nsubj → শিং *(sing)*]
- a lemma + suffix → surface form generation factor
  [i.e., উচেক *(uchek)* + শিং *(sing)* → উচেকশিং *(ucheksing)*]

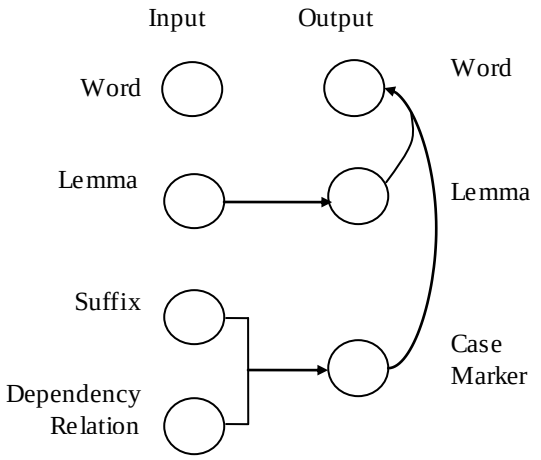


Figure 2. English to Manipuri translation factors

5.4 Factorization approach of Manipuri-English SMT system

Manipuri case markers are responsible to identify dependency relation and aspect information of English. Figure 3 shows the translation factors used in the translation between Manipuri and English. The Manipuri- English factorization consists of:

- **Translation factor:** lemma to lemma
  [e.g., উচেক (*uchek*) → Bird]
- **Translation factor:** suffix + POS → dependency relation + POS + suffix
  [e.g., শিং (*sing*) + NN → nsubj + NN + s]
- **Generation factor:** lemma + POS + dependency Relation +suffix → surface form generation factor
  [e.g., উচেক (*uchek*) + NN + nsubj + শিং (*sing*) → উচেকশিং (*ucheksing*]

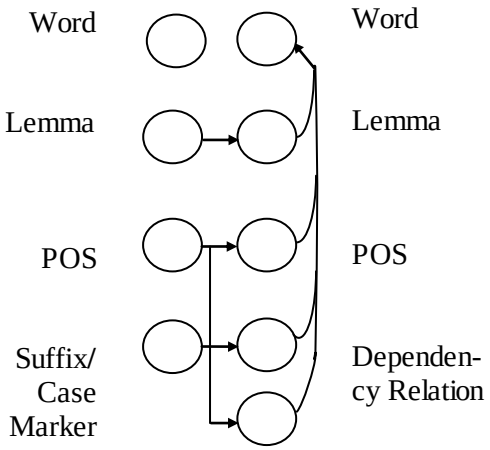


Figure 3. The Manipuri-English translation factors

5.5 Syntactically enriched output

High-order sequence models (just like n-gram language models over words) are used in order to support syntactic coherence of the output (Koehn and Hoang, 2007).

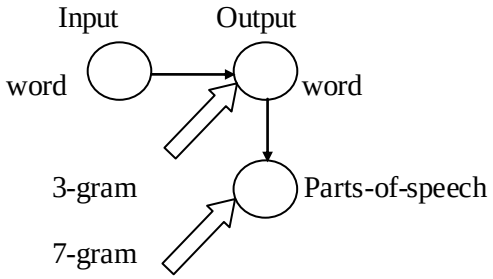


Figure 4. By generating additional linguistic factors on the output side, high-order sequence models over these factors support syntactical coherence of the output.

Adding part-of-speech factor on the output side and exploiting them with 7-gram sequence models (as shown in Figure 4) results in minor improvements in BLEU score.

6 Experimental Setup

A number of experiments have been carried out using factored translation framework and incorporating linguistic information. The toolkits used in the experiment are:

- *Stanford Dependency Parser*[4] was used to (i) generate the dependency relations and (ii) syntactic reordering of the input English sentences using Parse::RecDescent module.
- *Moses*[5] toolkit (Koehn, 2007) was used for training with GIZA++[6], decoding and minimum error rate training (Och, 2003) for tuning.
- *SRILM*[7] toolkit (Stolcke, 2002) was used to build language models with 10350 Manipuri sentences for English-Manipuri system and four and a half million English wordforms collected from the news domain for Manipuri-English system.
- English morphological analyzer *morpha*[8] (Minnen et al., 2001) was used and the

[4] http://nlp.stanford.edu/software/lex-parser.shtml
[5] http://www.statmt.org/moses/
[6] http://www.fjoch.com/GIZA++.html
[7] http://www.speech.sri.com/projects/srilm
[8] ftp://ftp.informatics.susx.ac.uk/pub/users/johnca/morph.tar.gz

stemmer from Manipuri Morphological analyzer (Singh and Bandyopadhyay, 2006) was used for the Manipuri side.

- *Manipuri POS tagger* (Singh et. al., 2008) is used to tag the POS (Parts of speech) factors of the input Manipuri sentences.

7 Evaluation

7.1 English-Manipuri SMT System

The evaluation of the machine translation systems developed in the present work is done in two approaches using automatic scoring with reference translation and subjective evaluation as discussed in (Ramanathan et al., 2009).

Evaluation Metrics:

- *NIST* (Doddington, 2002): A high score means a better translation by measuring the precision of n-gram.
- *BLEU* (Papineni et al, 2002): This metric gives the precision of n-gram with respect to the reference translation but with a brevity penalty.

	No of sentences	No of words
Training	10350	296728
Development	600	16520
Test	500	15204

Table 2. Training, development and testing corpus statistics

Table 2 shows the corpus statistics used in the experiment. The corpus is annotated with the proposed factors. The following models are developed for the experiment.

Baseline:

The model is developed using the default setting values in MOSES.

Lemma +Suffix:

It uses lemma and suffix factors on the source side, lemma and suffix on the target side for lemma to lemma and suffix to suffix translations with generation step of lemma plus suffix to surface form.

Lemma + Suffix + Dependency Relation:

Lemma, suffix and dependency relations are used on the source side. The translation steps are (a) lemma to lemma (b) suffix + dependency relation to suffix and generation step is lemma + suffix to surface form. Table 3 shows the BLEU and NIST scores of the system using these factors.

Table 4 shows the BLEU and NIST scores of the English-Manipuri SMT systems using lexicalized and syntactic reordering.

Model	BLEU	NIST
Baseline (surface)	13.045	4.25
Lemma + Suffix	15.237	4.79
Lemma + Suffix + Dependency Relation	16.873	5.10

Table 3. Evaluation Scores of English - Manipuri SMT System using various translation factors

Model	Reordering	BLEU	NIST
Baseline (surface)		13.045	4.25
Surface	Lexicalized	13.501	4.32
Surface	Syntactic	14.142	4.47

Table 4. Evaluation Scores of English-Manipuri SMT system using Lexicalized and Syntactic Reordering

Input/Output of English-Manipuri SMT:

(1a) **Input:** Going to school is obligatory for students.

স্কুল চৎপা ছাত্রশিংগী তৌদ য়াদ্রবা মখৌনি ।

School chatpa shatra-sing-gi touda ya draba mathouni.

Baseline output: স্কুল মখৌ চৎপা ওই ছাত্র

school mathou chatpa oy shatra

gloss : school duty going is student.

Syntactic Reorder output: ছাত্র স্কুল চৎপা তৌদ য়াদ্রবা

shatra school chatpa touda yadraba

gloss: Student school going compulsory.

Dependency output: ছাত্রশিং স্কুল চৎপা মখৌনি

shatrasing schoolda chatpa mathouni

gloss: Students going to the school is duty.

(1b) **Input:** Krishna has a flute in his hand.

কৃষ্ণগী খুত্তা তৌদ্রি অমা লৈ ।

Krishna-gi khut-ta toudri ama lei.

Syntactic Reorder output: কৃষ্ণ লৈ খুত্ অমা তৌদ্রি

Krishna lei khut ama toudri

gloss : Krishna has a hand flute

Dependency output: কৃষ্ণগী লৈ তৌদ্রি অমা খুত্তা

krishnagi lei toudri ama khutta

gloss : Krishna has a flute in his hand

One of the main aspects required for the fluency of a sentence is agreement. Certain words have to match in gender, case, number, person etc. within a sentence. The rules of agreement are language dependent and are closely linked to the morphological structure of language. Subjective evaluations on 100 sentences have been performed for fluency and adequacy by two judges. The fluency measures how well formed the sentences are at the output and adequacy measures the closeness of the output sentence with the reference translation. The Table 5 and Table 6 show the adequacy and fluency scales used for evaluation and Table 7 shows the scores of the evaluation.

Level	Interpretation
4	Full meaning is conveyed
3	Most of the meaning is conveyed
2	Poor meaning is conveyed
1	No meaning is conveyed

Table 5. Adequacy scale

Level	Interpretation
4	Flawless with no grammatical error
3	Good output with minor errors
2	Disfluent ungrammatical with correct phrase
1	Incomprehensible

Table 6. Fluency scale

	Sentence length	Fluency	Adequacy
Baseline	<=15 words	1.95	2.24
	>15 words	1.49	1.75
Reordered	<=15 words	2.58	2.75
	>15 words	1.82	1.96
Dependency Relation	<=15 words	2.83	2.91
	>15 words	1.94	2.10

Table 7. Scale of Fluency and Adequacy on sentence length basis of English-Manipuri SMT system

7.2 Manipuri-English SMT System

The system uses the corpus statistics shown in Table 2. The corpus is annotated with the proposed factors. The following models are developed for the experiment. The *baseline* and *lemma+suffix* systems follow same factors as English-Manipuri.

Lemma + Suffix + POS:
Lemma, suffix and POS are used on the source side. The translation steps are (a) lemma to lemma (b) suffix + POS to POS + suffix + dependency relation and generation step is lemma + suffix + POS + dependency relation to surface form.

Model	BLUE	NIST
Baseline (surface)	13.452	4.31
Lemma + Suffix	16.137	4.89
Lemma + Suffix + POS	17.573	5.15

Table 8. Evaluation Scores of Manipuri-English SMT system using various translation factors

Table 8 shows the BLEU and NIST scores of the Manipuri-English systems using the different factors. Table 9 shows the scores of using lexicalized reordering and POS language model.

Model	BLUE	NIST
Baseline + POS LM	14.341	4.52
Baseline + Lexicalized	13.743	4.46
Baseline + Lexicalized +POS LM	14.843	4.71

Table 9. Evaluation Scores of Manipuri-English SMT system using Lexicalized reordering and POS Language Model

Input/Output of Manipuri-English SMT:

(2a) **Input:** স্কুল চৎপা ছাত্রশিংগী তৌদ য়াদ্রবা মখৌনি |

gloss: *School chatpa shatra-sing-gi touda yadraba mathouni*.

Going to school is obligatory for students.

Baseline output: school going to the students important

Lexicalized Reordered output: school going important to the students

Lemma+Suffix+POS+lexicalized reordered output: School going important to the students

(2b) **Input:** কৃষ্ণগী খুত্তা তৌদ্রি অমা লৈ |

gloss: Krishna-gi khut-ta toudri ama lei.

Krishna has a flute in his hand.

Baseline output: Krishna is flute and hand

Lexicalized Reordered output: Krishna flute has his hand

Lemma+Suffix+POS+lexicalized reordered output: Krishna has flute his hand

By considering the lemma along with suffix and POS factors, the fluency and adequacy of the output is better addressed as given by the sample input and output (2a) and (2b) over the baseline system. Using the Manipuri stemmer, the case markers and suffixes are taken into account for different possible word forms thereby helping to overcome the data sparseness problem. Table 10 shows the scores of adequacy and fluency of the evaluation.

	Sentence length	Fluency	Adequacy
Baseline	<=15 words	1.93	2.31
	>15 words	1.51	1.76
Reordered	<=15 words	2.48	2.85
	>15 words	1.83	1.97
Lemma + Suffix + POS	<=15 words	2.86	2.92
	>15 words	2.01	2.11

Table 10. Scale of Fluency and Adequacy on sentence length basis of Manipuri-English SMT system

Subjective evaluations on 100 sentences have been performed for fluency and adequacy. In the process of subjective evaluation, sentences were judged on fluency, adequacy and the number of errors in case marking/morphology. It is observed that poor word-order makes the baseline output almost incomprehensible, while lexicalized reordering solves the problem correctly along with parts-of-speech language model (POS LM). Statistical significant test is performed to judge if a change in score that comes from a change in the system reflects a change in overall translation quality. It is found that all the differences are significant at the 99% level.

8 Discussion

The factored approach using the proposed factors show improved fluency and adequacy at the Manipuri output for English-Manipuri system as shown in the Table 6. Using the Stanford generated relations shows an improvement in terms of fluency and adequacy for shorter sentences than the longer ones.

Input : Khamba pushed the stone with a lever.
ꯈꯝꯅꯥ ꯖꯝꯐꯔꯝ ꯅꯨꯡ ꯑꯗꯨ ꯏꯂꯤꯡꯃꯤ |

Outputs:
Syntactic Reordered: ꯈꯝꯅ ꯅꯨꯡ ꯖꯝꯐꯗ꯭ ꯑꯗꯨ ꯏꯂꯤ |
Khamba nung jamfat adu illi
gloss: Khamba stone the lever push
Dependency: ꯈꯝꯅꯥ ꯅꯨꯡ ꯑꯗꯨ ꯖꯝꯐꯔꯝ ꯏꯂꯤ |
Khambana nung adu jamfatna illi
gloss: Khamba the stone pushed with lever

By the use of semantic relation, ꯥ (*na*) is attached to ꯈꯝꯅ (*Khamba*), which makes the meaning ꯈꯝꯅꯥ "by Khamba" instead of just ꯈꯝꯅ "Khamba".

Input : Suddenly the woman burst into tears.
ꯊꯥꯏꯍꯣꯗꯥ ꯃꯣ ꯑꯗꯨꯅ ꯃꯄꯤ ꯁꯤꯡꯔꯛꯀꯤ |

Outputs:
Syntactic Reordered: ꯅꯨꯄꯤ ꯊꯨꯅ ꯄꯤꯔꯥꯡꯒ꯭ꯒ ꯀꯦꯄꯤ |
Nupi thuna pirang-ga kappi
gloss: woman soon tears cry
Dependency: ꯑꯍꯨꯕꯗ ꯅꯨꯄꯤꯗ꯭ ꯀꯦꯄ꯭ꯂꯤ |
Athubada nupidu kaplammi
gloss: suddenly the woman cried

Here, in this example, the ꯅꯨꯄꯤ (nupi) is suffixed by the ꯗ꯭ (du), to produce ꯅꯨꯄꯤꯗ꯭ "the woman" instead of just ꯅꯨꯄꯤ "woman".

The factored approach of Manipuri-English SMT system also shows improved BLEU and NIST scores using the proposed factors as shown in Table 8 not only gain in fluency and adequacy scores as shown in Table 10.

9 Conclusion

A framework for Manipuri and English bidirectional SMT system using factored model is experimented with a goal to improve the translation output and reduce the amount of training data. The output of the translation is improved by incorporating morphological information and semantic relations by tighter integration. The systems are evaluated using automatic scoring techniques BLEU and NIST. The subjective evaluation of the systems is done to find out the fluency and adequacy. The fluency and adequacy are also addressed better for the shorter sentences than the longer ones using semantic relations. The improvement is statistically significant.

References

Avramidis, E. and Koehn, P. 2008. Enriching morphologically poor languages for Statistical Machine Translation, *Proceedings of ACL-08: HLT*

Callison-Burch, Chris., Osborne, M. and Koehn, P. 2006. Re-evaluating the Role of Bleu in Machine Translation Research" *In Proceedings of EACL-2006*

Doddington, G. 2002. Automatic evaluation of Machine Translation quality using n-gram co-occurrence statistics. *In Proceedings of HLT 2002*, San Diego, CA.

Koehn. P., and Hoang, H. 2007. Factored Translation Models, *In Proceedings of EMNLP-2007*

Koehn, P., Hieu, H., Alexandra, B., Chris, C., Marcello, F., Nicola, B., Brooke, C., Wade, S., Christine, M., Richard, Z., Chris, D., Ondrej, B., Alexandra, C., Evan, H. 2007. Moses: Open Source Toolkit for Statistical Machine Translation, *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 177–180, Prague.

Marie-Catherine de Marneffe and Manning, C. 2008. Stanford Typed Dependency Manual

Minnen, G., Carroll, J., and Pearce, D. 2001. Applied Morphological Processing of English, *Natural Language Engineering*, 7(3), pages 207-223

Nieβen, S., and Ney, H. 2004. Statistical Machine Translation with Scarce Resources Using Morpho-syntactic Information, *Computational Linguistics*, 30(2), pages 181-204

Och, F. 2003. Minimum error rate training in Statistical Machine Translation , *Proceedings of ACL*

Papineni, K., Roukos, S., Ward, T., and Zhu, W. 2002. BLEU: a method for automatic evaluation of machine translation. *In Proceedings of 40th ACL*, Philadelphia, PA

Popovic, M., and Ney, H. 2006. Statistical Machine Translation with a small amount of bilingual training data, *5th LREC SALTMIL Workshop on Minority Languages*

Ramanathan, A., Choudhury, H., Ghosh, A., and Bhattacharyya, P. 2009. Case markers and Morphology: Addressing the crux of the fluency problem in English-Hindi SMT, *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: 800-808

Rao, D., Mohanraj, K., Hegde, J., Mehta, V. and Mahadane, P. 2000. A practical framework for syntactic transfer of compound-complex sentences for English-Hindi Machine Translation, *Proceedings of KBCS* 2000, pages 343-354

Singh, Thoudam D., and Bandyopadhyay, S. 2006. Word Class and Sentence Type Identification in Manipuri Morphological Analyzer, *Proceeding of Modeling and Shallow Parsing of Indian Languages(MSPIL) 2006*, IIT Bombay, pages 11-17, Mumbai, India

Singh, Thoudam D., and Bandyopadhyay, S. 2008. Morphology Driven Manipuri POS Tagger, *In proceedings International Joint Conference on Natural Language Processing (IJCNLP-08) Workshop on Natural Language Processing of Less Privileged Languages (NLPLPL) 2008*, pages 91-98, Hyderabad, India

Singh, Thoudam D., and Bandyopadhyay, S. 2010a. Manipuri-English Example Based Machine Translation System, *International Journal of Computational Linguistics and Applications* (IJCLA), ISSN 0976-0962, pages 147-158

Singh, Thoudam D., Singh, Yengkhom R. and Bandyopadhyay, S., 2010b. Manipuri-English Semi Automatic Parallel Corpora Extraction from Web, *In proceedings of 23rd International Conference on the Computer Processing of Oriental Languages (ICCPOL 2010) - New Generation in Asian Information Processing* , San Francisco Bay, CA, USA, Pages 45-48

Singh, Thoudam D. and Bandyopadhyay, S., 2010c. Statistical Machine Translation of English-Manipuri using Morpho-Syntactic and Semantic Information, *In the proceedings of Ninth Conference of the Association for Machine Translation in Americas (AMTA 2010)*, Denver, Colorado, USA. (To appear)

Stolcke. A. 2002. SRILM - An Extensible Language Modeling Toolkit. *In Proc. Intl. Conf. Spoken Language Processing*, Denver, Colorado, September.

Wang, C., Collin, M., and Koehn, P. 2007. Chinese syntactic reordering for statistical machine translation, *Proceedings of EMNLP-CoNLL*

A Discriminative Syntactic Model for Source Permutation via Tree Transduction

Maxim Khalilov and **Khalil Sima'an**
Institute for Logic, Language and Computation
University of Amsterdam
{m.khalilov,k.simaan}@uva.nl

Abstract

A major challenge in statistical machine translation is mitigating the word order differences between source and target strings. While reordering and lexical translation choices are often conducted in tandem, *source string permutation* prior to translation is attractive for studying reordering using hierarchical and syntactic structure. This work contributes an approach for learning source string permutation via transfer of the source syntax tree. We present a novel discriminative, probabilistic tree transduction model, and contribute a set of empirical upperbounds on translation performance for English-to-Dutch source string permutation under sequence and parse tree constraints. Finally, the translation performance of our learning model is shown to outperform the state-of-the-art phrase-based system significantly.

1 Introduction

From its beginnings, statistical machine translation (SMT) has faced a word reordering challenge that has a major impact on translation quality. While standard mechanisms embedded in phrase-based SMT systems, e.g. (Och and Ney, 2004), deal efficiently with word reordering within a limited window of words, they are still not expected to handle all possible reorderings that involve words beyond this relatively narrow window, e.g., (Tillmann and Ney, 2003; Zens and Ney, 2003; Tillman, 2004). More recent work handles word order differences between source and target languages using hierarchical methods that draw on Inversion Transduction Grammar (ITG), e.g., (Wu and Wong, 1998; Chiang, 2005). In principle, the latter approach explores reordering defined by the choice of swapping the order of sibling subtrees under each node in a binary parse-tree of the source/target sentence.

An alternative approach aims at minimizing the need for reordering during translation by permuting the source sentence as a pre-translation step, e.g., (Collins et al., 2005; Xia and McCord, 2004; Wang et al., 2007; Khalilov, 2009). In effect, the translation process works with a model for source permutation ($s \rightarrow s^{'}$) followed by translation model ($s^{'} \rightarrow t$), where s and t are source and target strings and $s^{'}$ is the target-like permuted source string. In how far can source permutation reduce the need for reordering in conjunction with translation is an empirical question.

In this paper we define source permutation as the problem of learning how to *transfer* a given source parse-tree into a parse-tree that minimizes the divergence from target word-order. We model the tree transfer $\tau_s \rightarrow \tau_{s'}$ as a sequence of local, independent transduction operations, each transforming the current intermediate tree $\tau_{s'_i}$ into the next intermediate tree $\tau_{s'_{i+1}}$, with $\tau_{s_0} = \tau_s$ and $\tau_{s'_n} = \tau_{s'}$. A transduction operation merely permutes the sequence of $n > 1$ children of a single node in an intermediate tree, i.e., unlike previous work, we do not binarize the trees. The number of permutations is factorial in n, and learning a sequence of transductions for explaining a source permutation can be computationally rather challenging (see (Tromble and Eisner, 2009)). Yet,

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 92–100,
COLING 2010, Beijing, August 2010.

from the limited perspective of source *string* permutation ($s \rightarrow s^{'}$), another challenge is to integrate a figure of merit that measures in how far $s^{'}$ resembles a plausible target word-order.

We contribute solutions to these challenging problems. Firstly, we learn the transduction operations using a discriminative estimate of $P(\pi(\alpha_x) \mid N_x, \alpha_x, context_x)$, where N_x is the label of node (address) x, $N_x \rightarrow \alpha_x$ is the context-free production under x, $\pi(\alpha_x)$ is a permutation of α_x and $context_x$ represents a surrounding syntactic context. As a result, this constrains $\{\pi(\alpha_x)\}$ only to those found in the training data, and it conditions the transduction application probability on its specific contexts. Secondly, in every sequence $s_0^{'} = s, \ldots, s_n^{'} = s^{'}$ resulting from a tree transductions, we prefer those local transductions on $\tau_{s_{i-1}^{'}}$ that lead to source string permutation $s_i^{'}$ that are closer to target word order than $s_{i-1}^{'}$; we employ $s^{'}$ language model probability ratios as a measure of word order improvement.

In how far does the assumption of source permutation provide any window for improvement over a phrase-based translation system? We conduct experiments on translating from English into Dutch, two languages which are characterized by a number of systematic divergences between them. Initially, we conduct oracle experiments with varying constraints on source permutation to set upperbounds on performance relative to a state-of-the-art system. Translating the oracle source string permutation (obtained by untangling the crossing alignments) offers a large margin of improvement, whereas the oracle parse tree permutation provides a far smaller improvement. A minor change to the latter to also permute constituents that include words aligned with NULL, offers further improvement, yet lags bahind bare string permutation. Subsequently, we present translation results using our learning approach, and exhibit a significant improvement in BLEU score over the state-of-the-art baseline system. Our analysis shows that syntactic structure can provide important clues for reordering in translation, especially for dealing with long distance cases found in, e.g., English and Dutch. Yet, tree transduction by merely permuting the order of sister subtrees might turn out insufficient.

2 Baseline: Phrase-based SMT

Given a word-aligned parallel corpus, phrase-based systems (Och and Ney, 2002; Koehn et al., 2003) work with (in principle) arbitrarily large phrase pairs (also called blocks) acquired from word-aligned parallel data under a simple definition of translational equivalence (Zens et al., 2002). The conditional probabilities of one phrase given its counterpart are interpolated log-linearly together with a set of other model estimates:

$$\hat{e}_1^I = \arg\max_{e_1^I} \left\{ \sum_{m=1}^{M} \lambda_m h_m(e_1^I, f_1^J) \right\} \quad (1)$$

where a feature function h_m refer to a system model, and the corresponding λ_m refers to the relative weight given to this model. A phrase-based system employs feature functions for a phrase pair translation model, a language model, a reordering model, and a model to score translation hypothesis according to length. The weights λ_m are usually optimized for system performance (Och, 2003) as measured by BLEU (Papineni et al., 2002). Two reordering methods are widely used in phrase-based systems.

Distance-based A simple distance-based reordering model default for Moses system is the first reordering technique under consideration. This model provides the decoder with a cost linear to the distance between words that should be reordered.

MSD A lexicalized block-oriented data-driven reordering model (Tillman, 2004) considers three different orientations: monotone (M), swap (S), and discontinuous (D). The reordering probabilities are conditioned on the lexical context of each phrase pair, and decoding works with a block sequence generation process with the possibility of swapping a pair of blocks.

3 Related Work on Source Permutation

The integration of linguistic syntax into SMT systems offers a potential solution to reordering problem. For example, syntax is successfully integrated into hierarchical SMT (Zollmann and

Venugopal, 2006). Similarly, the tree-to-string syntax-based transduction approach offers a complete translation framework (Galley et al., 2006).

The idea of augmenting SMT by a reordering step prior to translation has often been shown to improve translation quality. Clause restructuring performed with hand-crafted reordering rules for German-to-English and Chinese-to-English tasks are presented in (Collins et al., 2005) and (Wang et al., 2007), respectively. In (Xia and McCord, 2004; Khalilov, 2009) word reordering is addressed by exploiting syntactic representations of source and target texts.

Other reordering models operate provide the decoder with multiple word orders. For example, the MaxEnt reordering model described in (Xiong et al., 2006) provides a hierarchical phrasal reordering system integrated within a CKY-style decoder. In (Galley and Manning, 2008) the authors present an extension of the famous MSD model (Tillman, 2004) able to handle long-distance word-block permutations. Coming up-to-date, in (PVS, 2010) an effective application of data mining techniques to syntax-driven source reordering for MT is presented.

Recently, Tromble and Eisner (2009) define source permutation as learning source permutations; the model works with a preference matrix for word pairs, expressing preference for their two alternative orders, and a corresponding weight matrix that is fit to the parallel data. The huge space of permutations is then structured using a binary synchronous context-free grammar (Binary ITG) with $O(n^3)$ parsing complexity, and the permutation score is calculated recursively over the tree at every node as the accumulation of the relative differences between the word-pair scores taken from the preference matrix. Application to German-to-English translation exhibits some performance improvement.

Our work is in the general learning direction taken in (Tromble and Eisner, 2009) but differs both in defining the space of permutations, using local probabilistic tree transductions, as well as in the learning objective aiming at scoring permutations based on a log-linear interpolation of a local syntax-based model with a global string-based (language) model.

4 Pre-Translation Source Permutation

Given a word-aligned parallel corpus, we define the source string permutation as the task of learning to unfold the crossing alignments between sentence pairs in the parallel corpus. Let be given a source-target sentence pair $s \rightarrow t$ with word alignment set a between their words. Unfolding the crossing instances in a should lead to as monotone an alignment $a^{'}$ as possible between a permutation $s^{'}$ of s and the target string t. Conducting such a "monotonization" on the parallel corpus gives two parallel corpora: (1) a source-to-permutation parallel corpus ($s \rightarrow s^{'}$) and (2) a source permutation-to-target parallel corpus ($s^{'} \rightarrow t$). The latter corpus is word-aligned automatically again and used for training a phrase-based translation system, while the former corpus is used for training our model for pre-translation source permutation via parse tree transductions.

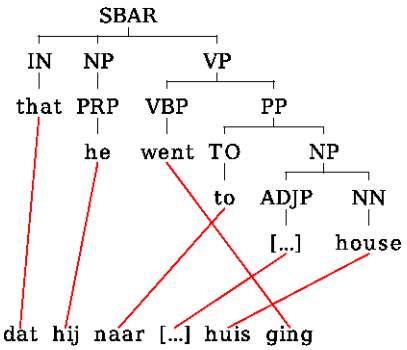
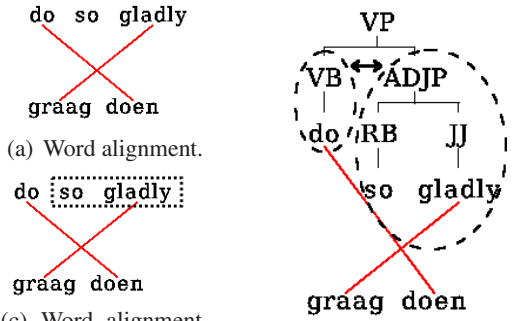


Figure 1: Example of crossing alignments and long-distance reordering using a source parse tree.

In itself, the problem of permuting the source string to unfold the crossing alignments is computationally intractable (see (Tromble and Eisner, 2009)). However, different kinds of constraints can be made on unfolding the crossing alignments in a. A common approach in hierarchical SMT is to assume that the source string has a binary parse tree, and the set of eligible permutations is defined by binary ITG transductions on this tree. This defines permutations that can be obtained only by at most inverting pairs of children under nodes of the source tree. Figure 1 exhibits a long distance reordering of the verb in English-to-Dutch translation: inverting the order of the children under the VP node would unfold the crossing alignment.

4.1 Oracle Performance

As has been shown in the literature (Costa-jussà and Fonollosa, 2006; Khalilov and Sima'an, 2010; Wang et al., 2007), source and target texts monotonization leads to a significant improvement in terms of translation quality. However it is not known how many alignment crossings can be unfolded under different parse tree conditions. In order to gauge the impact of corpus monotonization on translation system performance, we trained a set of oracle translation systems, which create target sentences that follow the source language word order using the word alignment links and various constraints.



(a) Word alignment.

(c) Word alignment and ADJP span.

(b) Parse tree and corresponding alignment.

Figure 2: Reordering example.

The set-up of our experiments and corpus characteristics are detailed in Section 5. Table 1 reports translation scores of the oracle systems. Notice that all the numbers are calculated on the realigned corpora. Baseline results are provided for informative purposes.

String permutation The first oracle system under consideration is created by traversing the string from left to right and unfolding all crossing alignment links (we call this system *oracle-string*). For example in Figure 2(a), the *oracle-string* system generates a string "`do so gladly`" swapping the words "`do`" and "`gladly`" without considering the parse tree. The first line of the table shows the performance of the *oracle-string* system with monotone source and target portions of the corpus.

Oracle under tree constraint We use a syntactic parser for parsing the English source sentences that provide n-ary constituency parses. Now we constrain unfolding crossing alignments only to those alignment links which agree with the structure of the source-side parse tree and consider the constituents which include aligned tokens only. Unfolding a crossing alignment is modeled as permuting the children of a node in the parse tree. We refer to this oracle system as *oracle-tree*. For example provided in Figure 2(b), there is no way to construct a monotonized version of the sentence since the word "`so`" is aligned to NULL and impedes swapping the order of VB and ADJP under the VP.

Oracle under relaxed tree constraint The *oracle-tree* system does not permute the words which are both (1) not found in the alignment and (2) are spanned by the sub-trees sibling to the reordering constituents. Now we introduce a relaxed version of the parse tree constraint: the order of the children of a node is permuted when the node covers the reordering constituents and *also* when the frontier contains leaf nodes aligned with NULL (*oracle-span*). For example, in Figure 2(c) the English word "`so`" is not aligned, but according to the relaxed version, must move together with the word "`gladly`" since they share a parent node (*ADJP*).

Source	BLEU	NIST
baseline dist	24.04	6.29
baseline MSD	24.04	6.28
oracle − string	27.02	6.51
oracle − tree	24.09	6.30
oracle − span	24.95	6.37

Table 1: Translation scores of oracle systems.

The main conclusion which can be drawn from the oracle results is that there is a possibility for relatively big (≈3 BLEU points) improvement with complete unfolding of crossing alignments and very limited (≈0.05 BLEU points) with the same done under the parse tree constraint. A tree-based system that allows for permuting unaligned words that are covered by a dominating parent node shows more improvement in terms of BLEU and NIST scores (≈0.9 BLEU points).

The gap between *oracle-string* and *oracle-tree* performance is due to alignment crossings which

cannot be unfolded under trees (illustrated in Figure 3), but possibly also due to parse and alignment errors.

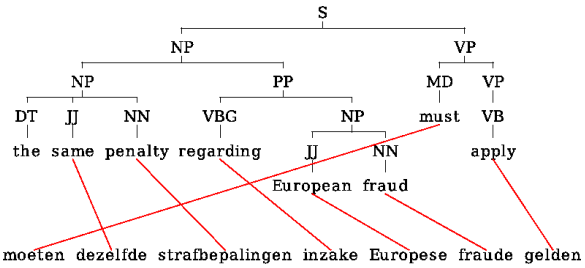


Figure 3: Example of alignment crossing that does not agree with the parse tree.

4.2 Source Permutation via Syntactic Transfer

Given a parallel corpus with string pairs $s \rightarrow t$ with word alignment a, we create a *source permuted* parallel corpus $s \rightarrow s'$ by unfolding the crossing alignments in a: this is done by scanning the string s from left to right and moving words involved in crossing alignments to positions where the crossing alignments are unfolded). The source strings s are parsed, leading to a single parse tree τ_s per source string.

Our model aims at learning from the source permuted parallel corpus $s \rightarrow s'$ a probabilistic optimization $\arg\max_{\pi(s)} P(\pi(s) \mid s, \tau_s)$. We assume that the set of permutations $\{\pi(s)\}$ is defined through a finite set of local transductions over the tree τ_s. Hence, we view the permutations leading from s to s' as a sequence of local tree transductions $\tau_{s'_0} \rightarrow \ldots \rightarrow \tau_{s'_n}$, where $s'_0 = s$ and $s'_n = s'$, and each transduction $\tau_{s'_{i-1}} \rightarrow \tau_{s'_i}$ is defined using a tree transduction operation that *at most permutes the children of a single node in* $\tau_{s'_{i-1}}$ *as defined next*.

A local transduction $\tau_{s'_{i-1}} \rightarrow \tau_{s'_i}$ is modelled by an operation that applies to a single node with address x in $\tau_{s'_{i-1}}$, labeled N_x, and may permute the ordered sequence of children α_x dominated by node x. This constitutes a direct generalization of the ITG binary inversion transduction operation. We assign a conditional probability to each such local transduction:

$$P(\tau_{s'_i} \mid \tau_{s'_{i-1}}) \approx P(\pi(\alpha_x) \mid N_x \rightarrow \alpha_x, C_x) \quad (2)$$

where $\pi(\alpha_x)$ is a permutation of α_x (the ordered sequence of node labels under x) and C_x is a local tree context of node x in tree $\tau_{s'_{i-1}}$. One wrinkle in this definition is that the number of possible permutations of α_x is factorial in the length of α_x. Fortunately, the source permuted training data exhibits only a fraction of possible permutations even for longer α_x sequences. Furthermore, by conditioning the probability on local context, the general applicability of the permutation is restrained.

Given this definition, we define the probability of the sequence of local tree transductions $\tau_{s'_0} \rightarrow \ldots \rightarrow \tau_{s'_n}$ as

$$P(\tau_{s'_0} \rightarrow \ldots \rightarrow \tau_{s'_n}) = \prod_{i=1}^{n} P(\tau_{s'_i} \mid \tau_{s'_{i-1}}) \quad (3)$$

The problem of calculating the most likely permutation under this transduction model is made difficult by the fact that different transduction sequences may lead to the same permutation, which demands summing over these sequences. Furthermore, because every local transduction conditions on local context of an intermediate tree, this quickly risks becoming intractable (even when we use packed forests). In practice we take a pragmatic approach and greedily select at every intermediate point $\tau_{s'_{i-1}} \rightarrow \tau_{s'_i}$ the single most likely local transduction that can be conducted on any node of the current intermediate tree $\tau_{s'_{i-1}}$ using an interpolation of the term in Equation 2 with string probability ratios as follows:

$$P(\pi(\alpha_x) \mid N_x \rightarrow \alpha_x, C_x) \times \frac{P(s'_{i-1})}{P(s'_i)}$$

The rationale behind this log-linear interpolation is that our source permutation approach aims at finding the optimal permutation s' of s that can serve as input for a subsequent translation model. Hence, we aim at tree transductions that are syntactically motivated that also lead to improved string permutation. In this sense, the tree transduction definitions can be seen as an efficient and

syntactically informed way to define the space of possible permutations.

We estimate the string probabilities $P(s_i^{'})$ using 5-gram language models trained on the $s^{'}$ side of the source permuted parallel corpus $s \rightarrow s^{'}$. We estimate the conditional probability $P(\pi(\alpha_x) \mid N_x \rightarrow \alpha_x, C_x)$ using a Maximum-Entropy framework, where feature functions are defined to capture the permutation as a class, the node label N_x and its head POS tag, the child sequence α_x together with the corresponding sequence of head POS tags and other features corresponding to different contextual information.

We were particularly interested in those linguistic features that motivate reordering phenomena from the syntactic and linguistic perspective. The features that were used for training the permutation system are extracted for every internal node of the source tree that has more than one child:

- *Local tree topology.* Sub-tree instances that include parent node and the ordered sequence of child node labels.
- *Dependency features.* Features that determine the POS tag of the head word of the current node, together with the sequence of POS tags of the head words of its child nodes.
- *Syntactic features.* Three binary features from this class describe: (1) whether the parent node is a child of the node annotated with the same syntactic category, (2) whether the parent node is a descendant of the node annotated with the same syntactic category, and (3) if the current subtree is embedded into a "*SENT-SBAR*" sub-tree. The latter feature intends to model the divergence in word order in relative clauses between Dutch and English which is illustrated in Figure 1.

In initial experiments we piled up all feature functions into a single model. Preliminary results showed that the system performance increases if the set of patterns is split into partial classes conditioned on the current node label. Hence, we trained four separate MaxEnt models for the categories with potentially high number of crossing alignments, namely *VP*, *NP*, *SENT*, and *SBAR*. For combinatory models we use the following notations: $M_4 = \sum_{i \in [\text{ NP, VP, SENT, SBAR}]} M_i$ and $M_2 = \sum_{i \in [\text{VP, SENT}]} M_i$.

5 Experiments and results

The SMT system used in the experiments was implemented within the open-source MOSES toolkit (Koehn et al., 2007). Standard training and weight tuning procedures which were used to build our system are explained in details on the MOSES web page[1]. The MSD model was used together with a distance-based reordering model. Word alignment was estimated with GIZA++ tool[2] (Och, 2003), coupled with mkcls[3] (Och, 1999), which allows for statistical word clustering for better generalization. An 5-gram target language model was estimated using the SRI LM toolkit (Stolcke, 2002) and smoothed with modified Kneser-Ney discounting. We use the Stanford parser[4] (Klein and Manning, 2003) as a source-side parsing engine. The parser was trained on the English treebank set provided with 14 syntactic categories and 48 POS tags. The evaluation conditions were case-sensitive and included punctuation marks. For Maximum Entropy modeling we used the maxent toolkit[5].

Data The experiment results were obtained using the English-Dutch corpus of the European Parliament Plenary Session transcription (*EuroParl*). Training corpus statistics can be found in Table 2.

	Dutch	English
Sentences	1.2 M	1.2 M
Words	32.9 M	33.0 M
Average sentence length	27.20	27.28
Vocabulary	228 K	104 K

Table 2: Basic statistics of the English-Dutch EuroParl training corpus.

The development and test datasets were randomly chosen from the corpus and consisted of

[1] http://www.statmt.org/moses/
[2] code.google.com/p/giza-pp/
[3] http://www.fjoch.com/mkcls.html
[4] http://nlp.stanford.edu/software/lex-parser.shtml
[5] http://homepages.inf.ed.ac.uk/lzhang10/maxent_toolkit.html

500 and 1,000 sentences, respectively. Both were provided with one reference translation.

Results Evaluation of the system performance is twofold. In the first step, we analyze the quality of reordering method itself. In the next step we look at the automatic translation scores and evaluate the impact which the choice of reordering strategy has on the translation quality. In both stages of evaluation, the results are contrasted with the performance shown by the standard phrase-based SMT system (*baseline*) and with oracle results.

Source reordering analysis Table 3 shows the parameters of the reordered system allowing to assess the effectiveness of reordering permutations, namely: (1) a total number of crossings found in the word alignment (#C), (2) the size of the resulting phrase table (PT), (3) BLEU, NIST, and WER scores obtained using monotonized parallel corpus (*oracle*) as a reference.

All the numbers are calculated on the re-aligned corpora. Calculations are done on the basis of the 100,000 line extraction from the corpus[6] and corresponding alignment matrix. The *baseline* rows show the number of alignment crossings found in the original (unmonotonized) corpus.

System	#C	PT	Scores		
			BLEU	NIST	WER
Oracle					
string	54.6K	48.4M	-	-	-
tree	187.3K	30.3M	71.73	17.01	16.77
span	146.9K	33.0M	73.41	17.11	15.73
Baselines					
baselines	187.0K	29.8M	71.70	17.07	16.55
Category models					
M_{NP}	188.9K	29.7M	71.63	17.07	16.52
M_{VP}	168.1K	29.8M	73.17	17.16	15.99
M_{SENT}	171.0K	29.8M	73.08	17.08	16.10
M_{SBAR}	188.6K	29.8M	72.89	16.90	16.41
Combinatory models					
M_4	193.2K	29.1M	70.98	16.85	16.78
M_2	165.4K	29.9M	73.07	16.92	15.88

Table 3: Main parameters of the tree-based reordering system.

[6]A smaller portion of the corpus is used for analysis in order to reduce evaluation time.

Translation scores The evaluation results for the development and test corpora are reported in Table 4. They include two *baseline* configurations (*dist* and *MSD*), *oracle* results and contrasts them with the performance shown by different combinations of single-category tree-based reordering models. Best scores within each experimental section are placed in cells filled with grey.

System	Dev	Test	
	BLEU	BLEU	NIST
baseline dist	23.88	24.04	6.29
baseline MSD	24.07	24.04	6.28
oracle-string	26.28	27.02	6.50
oracle-tree	23.84	24.09	6.30
oracle-span	24.79	24.95	6.35
M_{NP}	23.79	23.81	6.27
M_{VP}	24.16	24.55	6.29
M_{SENT}	24.27	24.56	6.32
M_{SBAR}	23.99	24.12	6.27
M_4	23.50	23.86	6.29
M_2	24.28	24.64	6.33

Table 4: Experimental results.

Analysis The number of crossings found in word alignment intersection and BLEU/NIST/WER scores estimated on reordered data vs. monotonized data report the reordering algorithm effectiveness. A big gap between number of crossings and total number of reorderings per corpus found in *oracle-string* system[7] and *baseline* systems demonstrates the possible reduction of system's non-monotonicity. The difference in number of crossings and BLEU/NIST/WER scores between the *oracle-span* and the best performing MaxEnt models (namely, M_2) shows the level of performance of the prediction module.

A number of distinct phrase translation pairs in the translation table implicitly reveals the generalization capabilities of the translation system since it simplifies the translation task. From the other hand, increased number of shorter phrases can add noise in the reordered data and makes decoding more complex. Hence, the size of phrase table itself can not be considered as a robust indicator of its translation potential.

[7]The number of crossings for *oracle* configuration is not zero since this parameter is calculated on the re-aligned corpus.

Table 4 shows that three of six MaxEnt reordering systems outperform *baseline* systems by about 0.5-0.6 BLEU points, that is statistically significant[8]. The combination of *NP*, *NP*, *SENT*, and *SBAR* models do not show good performance possibly due to increased sparseness of reordering patterns. However, the system that consider only the M_{VP} and M_{SENT} models achieves 0.62 BLEU score gain over the *baseline* configurations.

The main conclusion which can be drawn from analysis of Tables 3 and 4 is that there is an evident correlation between characteristics of reordering system and performance demonstrated by the translation system trained on the corpus with reordered source part.

Example Figure 4 exemplifies the sentences that presumably benefits from the monotonization of the source part of the parallel corpus. The example demonstrates a pervading syntactic distinction between English and Dutch: the reordering of verb-phrase subconstituents *VP NP PP* within the relative clause into *PP NP VP*.

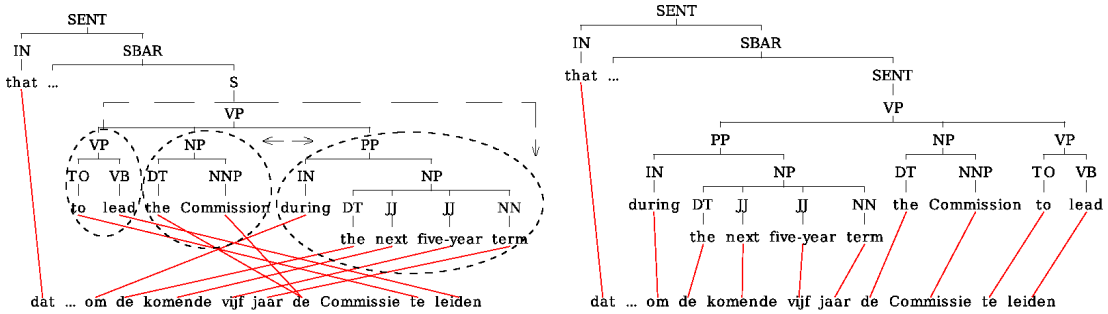
6 Conclusions and future work

We introduced a tree-based reordering model that aims at monotonizing the word order of source and target languages as a pre-translation step. Our model avoids complete generalization of reordering instances by using tree contexts and limiting the permutations to data instances. From a learning perspective, our work shows that navigating a large space of intermediate tree transformations can be conducted effectively using both the source-side syntactic tree and a language model of the idealized (target-like) source-permuted language.

We have shown the potential for translation quality improvement when target sentences are created following the source language word order (≈3 BLEU points over the standard phrase-based SMT) and under parse tree constraint (≈0.9 BLEU points). As can be seen from these results, our model exhibits competitive translation performance scores compared with the standard distance-based and lexical reordering.

The gap between the oracle and our system's results leaves room for improvement. We intend to study extensions of the current tree transfer model to narrow this performance gap. As a first step we are combining isolated models for concrete syntactic categories and aggregating more features into the MaxEnt model. Algorithmic improvements, such as beam-search and chart parsing, could allow us to apply our method to full parse-forests as opposed to a single parse tree.

[8] All statistical significance calculations are done for a 95% confidence interval and 1 000 resamples, following the guidelines from (Koehn, 2004).



(a) Original parse tree.

(b) Reordered parse tree.

Src: *that ... to lead the Commission during the next five-year term*
Ref.: *dat ... om de komende vijf jaar de Commissie te leiden*
Baseline MSD: *dat ... om het voortouw te nemen in de Commissie tijdens de komende vijf jaar*
Rrd src: *that ... during the next five-year term the Commission to lead*
$\mathbf{M_2}$: *dat ... om de Commissie tijdens de komende vijf jaar te leiden*

(c) Translations.

Figure 4: Example of tree-based monotonization.

References

D. Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of ACL'05*, pages 263–270.

M. Collins, P. Koehn, and I. Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of ACL'05*, pages 531–540.

M. R. Costa-jussà and J. A. R. Fonollosa. 2006. Statistical machine reordering. In *Proceedings of HLT/EMNLP'06*, pages 70–76.

M. Galley and Ch. D. Manning. 2008. A simple and effective hierarchical phrase reordering model. In *Proceedings of EMNLP'08*, pages 848–856.

M. Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang, and I. Thaye. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proc. of COLING/ACL'06*, pages 961–968.

M. Khalilov and K. Sima'an. 2010. Source reordering using maxent classifiers and supertags. In *Proc. of EAMT'10*, pages 292–299.

M. Khalilov. 2009. *New statistical and syntactic models for machine translation*. Ph.D. thesis, Universitat Politècnica de Catalunya, October.

D. Klein and C. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the ACL'03*, pages 423–430.

Ph. Koehn, F. Och, and D. Marcu. 2003. Statistical phrase-based machine translation. In *Proceedings of the HLT-NAACL 2003*, pages 48–54.

Ph. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. 2007. Moses: open-source toolkit for statistical machine translation. In *Proceedings of ACL 2007*, pages 177–180.

Ph. Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP'04*, pages 388–395.

F. Och and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of ACL'02*, pages 295–302.

F. Och and H. Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.

F. Och. 1999. An efficient method for determining bilingual word classes. In *Proceedings of ACL 1999*, pages 71–76.

F. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL'03*, pages 160–167.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of ACL'02*, pages 311–318.

A. PVS. 2010. A data mining approach to learn reorder rules for SMT. In *Proceedings of NAACL/HLT'10*, pages 52–57.

A. Stolcke. 2002. SRILM: an extensible language modeling toolkit. In *Proceedings of SLP'02*, pages 901–904.

C. Tillman. 2004. A unigram orientation model for statistical machine translation. In *Proceedings of HLT-NAACL'04*, pages 101–104.

C. Tillmann and H. Ney. 2003. Word reordering and a dynamic programming beam search algorithm for statistical machine translation. *Computational Linguistics*, 1(29):93–133.

R. Tromble and J. Eisner. 2009. Learning linear ordering problems for better translation. In *Proceedings of EMNLP'09*, pages 1007–1016.

C. Wang, M. Collins, and Ph. Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of EMNLP-CoNLL'07*, pages 737–745.

D. Wu and H. Wong. 1998. Machine translation wih a stochastic grammatical channel. In *Proceedings of ACL-COLING'98*, pages 1408–1415.

F. Xia and M. McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *Proceedings of COLING'04*, pages 508–514.

D. Xiong, Q. Liu, and S. Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of ACL'06*, pages 521–528.

R. Zens and H. Ney. 2003. A comparative study on reordering constraints in statistical machine translation. In *Proceedings of ACL'03*, pages 144–151.

R. Zens, F. Och, and H. Ney. 2002. Phrase-based statistical machine translation. In *Proceedings of KI: Advances in Artificial Intelligence*, pages 18–32.

A. Zollmann and A. Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings of NAACL'06*, pages 138–141.

HMM Word-to-Phrase Alignment with Dependency Constraints

Yanjun Ma **Andy Way**
Centre for Next Generation Localisation
School of Computing
Dublin City University
{yma,away}@computing.dcu.ie

Abstract

In this paper, we extend the HMM word-to-phrase alignment model with syntactic dependency constraints. The syntactic dependencies between multiple words in one language are introduced into the model in a bid to produce coherent alignments. Our experimental results on a variety of Chinese–English data show that our syntactically constrained model can lead to as much as a 3.24% relative improvement in BLEU score over current HMM word-to-phrase alignment models on a Phrase-Based Statistical Machine Translation system when the training data is small, and a comparable performance compared to IBM model 4 on a Hiero-style system with larger training data. An intrinsic alignment quality evaluation shows that our alignment model with dependency constraints leads to improvements in both precision (by 1.74% relative) and recall (by 1.75% relative) over the model without dependency information.

1 Introduction

Generative word alignment models including IBM models (Brown et al., 1993) and HMM word alignment models (Vogel et al., 1996) have been widely used in various types of Statistical Machine Translation (SMT) systems. This widespread use can be attributed to their robustness and high performance particularly on large-scale translation tasks. However, the quality of the alignment yielded from these models is still far from satisfactory even with significant amounts of training data; this is particularly true for radically different languages such as Chinese and English.

The weakness of most generative models often lies in the incapability of addressing one to many (1-to-n), many to one (n-to-1) and many to many (m-to-n) alignments. Some research directly addresses m-to-n alignment with phrase alignment models (Marcu and Wong, 2002). However, these models are unsuccessful largely due to intractable estimation (DeNero and Klein, 2008). Recent progress in better parameterisation and approximate inference (Blunsom et al., 2009) can only augment the performance of these models to a similar level as the baseline where bidirectional word alignments are combined with heuristics and subsequently used to induce translation equivalence (e.g. (Koehn et al., 2003)). The most widely used word alignment models, such as IBM models 3 and 4, can only model 1-to-n alignment; these models are often called "asymmetric" models. IBM models 3 and 4 model 1-to-n alignments using the notion of "fertility", which is associated with a "deficiency" problem despite its high performance in practice.

On the other hand, the HMM word-to-phrase alignment model tackles 1-to-n alignment problems with simultaneous segmentation and alignment while maintaining the efficiency of the models. Therefore, this model sets a good example of addressing the tradeoffs between modelling power and modelling complexity. This model can also be seen as a more generalised

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 101–109,
COLING 2010, Beijing, August 2010.

case of the HMM word-to-word model (Vogel et al., 1996; Och and Ney, 2003), since this model can be reduced to an HMM word-to-word model by restricting the generated target phrase length to one. One can further refine existing word alignment models with syntactic constraints (e.g. (Cherry and Lin, 2006)). However, most research focuses on the incorporation of syntactic constraints into discriminative alignment models. Introducing syntactic information into generative alignment models is shown to be more challenging mainly due to the absence of appropriate modelling of syntactic constraints and the "inflexibility" of these generative models.

In this paper, we extend the HMM word-to-phrase alignment model with syntactic dependencies by presenting a model that can incorporate syntactic information while maintaining the efficiency of the model. This model is based on the observation that in 1-to-n alignments, the n words bear some syntactic dependencies. Leveraging such information in the model can potentially further aid the model in producing more fine-grained word alignments. The syntactic constraints are specifically imposed on the n words involved in 1-to-n alignments, which is different from the cohesion constraints (Fox, 2002) as explored by Cherry and Lin (2006), where knowledge of cross-lingual syntactic projection is used. As a syntactic extension of the open-source MTTK implementation (Deng and Byrne, 2006) of the HMM word-to-phrase alignment model, its source code will also be released as open source in the near future.

The remainder of the paper is organised as follows. Section 2 describes the HMM word-to-phrase alignment model. In section 3, we present the details of the incorporation of syntactic dependencies. Section 4 presents the experimental setup, and section 5 reports the experimental results. In section 6, we draw our conclusions and point out some avenues for future work.

2 HMM Word-to-Phrase Alignment Model

In HMM word-to-phrase alignment, a sentence **e** is segmented into a sequence of consecutive phrases: $\mathbf{e} = v_1^K$, where v_k represents the k^{th} phrase in the target sentence. The assumption that each phrase v_k generated as a translation of one single source word is consecutive is made to allow efficient parameter estimation. Similarly to word-to-word alignment models, a variable a_1^K is introduced to indicate the correspondence between the target phrase index and a source word index: $k \rightarrow i = a_k$ indicating a mapping from a target phrase v_k to a source word f_{a_k}. A random process ϕ_k is used to specify the number of words in each target phrase, subject to the constraints $J = \sum_{k=1}^{K} \phi_k$, implying that the total number of words in the phrases agrees with the target sentence length J.

The insertion of target phrases that do not correspond to any source words is also modelled by allowing a target phrase to be aligned to a non-existent source word f_0 (NULL). Formally, to indicate whether each target phrase is aligned to NULL or not, a set of indicator functions $\varepsilon_1^K = \{\varepsilon_1, \cdots, \varepsilon_K\}$ is introduced (Deng and Byrne, 2008): if $\varepsilon_k = 0$, then NULL $\rightarrow v_k$; if $\varepsilon_k = 1$, then $f_{a_k} \rightarrow v_k$.

To summarise, an alignment **a** in an HMM word-to-phrase alignment model consists of the following elements:

$$\mathbf{a} = (K, \phi_1^K, a_1^K, \varepsilon_1^K)$$

The modelling objective is to define a conditional distribution $P(\mathbf{e}, \mathbf{a}|\mathbf{f})$ over these alignments. Following (Deng and Byrne, 2008), $P(\mathbf{e}, \mathbf{a}|\mathbf{f})$ can be decomposed into a phrase count distribution (1) modelling the segmentation of a target sentence into phrases ($P(K|J, \mathbf{f}) \propto \eta^K$ with scalar η to control the length of the hypothesised phrases), a transition distribution (2) modelling the dependencies between the current link and the previous links, and a word-to-phrase translation distribution (3) to model the degree to which a word and a phrase are translational to each other.

$$\begin{aligned} P(\mathbf{e}, \mathbf{a}|\mathbf{f}) &= P(v_1^K, K, a_1^K, \varepsilon_1^K, \phi_1^K|\mathbf{f}) \\ &= P(K|J, \mathbf{f}) \quad (1) \\ &\quad P(a_1^K, \varepsilon_1^K, \phi_1^K|K, J, \mathbf{f}) \quad (2) \\ &\quad P(v_1^K|a_1^K, \varepsilon_1^K, \phi_1^K, K, J, \mathbf{f}) \quad (3) \end{aligned}$$

The **word-to-phrase translation distribution** (3) is formalised as in (4):

$$P(v_1^K|a_1^K, \varepsilon_1^K, \phi_1^K, K, J, \mathbf{f}) = \prod_{k=1}^{K} p_v(v_k|\varepsilon_k \cdot f_{a_k}, \phi_k) \quad (4)$$

Note here that we assume that the translation of each target phrase is conditionally independent of other target phrases given the individual source words.

If we assume that each word in a target phrase is translated with a dependence on the previously translated word in the same phrase given the source word, we derive the bigram translation model as follows:

$$p_v(v_k|f_{a_k}, \varepsilon_k, \phi_k) = p_{t_1}(v_k[1]|\varepsilon_k, f_{a_k}) \prod_{j=2}^{\phi_k} p_{t_2}(v_k[j]|v_k[j-1], \varepsilon_k, f_{a_k})$$

where $v_k[1]$ is the first word in phrase v_k, $v_k[j]$ is the j^{th} word in v_k, p_{t_1} is an unigram translation probability and p_{t_2} is a bigram translation probability. The intuition is that the first word in v_k is firstly translated by f_{a_k} and the translation of the remaining words $v_k[j]$ in v_k from f_{a_k} is dependent on the translation of the previous word $v_k[j-1]$ from f_{a_k}. The use of a bigram translation model can address the coherence of the words within the phrase v_k so that the quality of phrase segmentation can be improved.

3 Syntactically Constrained HMM Word-to-Phrase Alignment Models

3.1 Syntactic Dependencies for Word-to-Phrase Alignment

As a proof-of-concept, we performed dependency parsing on the GALE gold-standard word alignment corpus using Maltparser (Nivre et al., 2007).[1] We find that 82.54% of the consecutive English words have syntactic dependencies and 77.46% non-consecutive English words have syntactic dependencies in 1-to-2 Chinese–English (ZH–EN) word alignment (one Chinese word aligned to two English words). For English–Chinese (EN–ZH) word alignment, we observe that 75.62% of the consecutive Chinese words and 71.15% of the non-consecutive Chinese words have syntactic dependencies. Our model represents an attempt to encode these linguistic intuitions.

[1] http://maltparser.org/

3.2 Component Variables and Distributions

We constrain the word-to-phrase alignment model with a syntactic coherence model. Given a target phrase v_k consisting of ϕ_k words, we use the dependency label r_k between words $v_k[1]$ and $v_k[\phi_k]$ to indicate the level of coherence. The dependency labels are a closed set obtained from dependency parsers, e.g. using Maltparser, we have 20 dependency labels for English and 12 for Chinese in our data. Therefore, we have an additional variable r_1^K associated with the sequence of phrases v_1^K to indicate the syntactic coherence of each phrase, defining $P(\mathbf{e}, \mathbf{a}|\mathbf{f})$ as below:

$$P(r_1^K, v_1^K, K, a_1^K, \varepsilon_1^K, \phi_1^K|\mathbf{f}) = P(K|J, \mathbf{f}) P(a_1^K, \phi_1^K, \varepsilon_1^K|K, J, \mathbf{f}) P(v_1^K|a_1^K, \varepsilon_1^K, \phi_1^K, K, J, \mathbf{f}) P(r_1^K|a_1^K, \varepsilon_1^K, \phi_1^K, v_1^K, K, J, \mathbf{f}) \quad (5)$$

The **syntactic coherence distribution** (5) is simplified as in (6):

$$P(r_1^K|a_1^K, \varepsilon_1^K, \phi_1^K, v_1^K, K, J, \mathbf{f}) = \prod_{k=1}^{K} p_r(r_k; \varepsilon, f_{a_k}, \phi_k) \quad (6)$$

Note that the coherence of each target phrase is conditionally independent of the coherence of other target phrases given the source words f_{a_k} and the number of words in the current phrase ϕ_k. We name the model in (5) the SSH model. SSH is an abbreviation of Syntactically constrained Segmental HMM, given the fact that the HMM word-to-phrase alignment model is a Segmental HMM model (SH) (Ostendorf et al., 1996; Murphy, 2002).

As our syntactic coherence model utilises syntactic dependencies which require the presence of at least two words in target phrase v_k, we therefore model the cases of $\phi_k = 1$ and $\phi_k \geq 2$

separately. We rewrite (6) as follows:

$$p_r(r_k;\varepsilon,f_{a_k},\phi_k) = \begin{cases} p_{\phi_k=1}(r_k;\varepsilon,f_{a_k}) & \text{if } \phi_k = 1 \\ p_{\phi_k\geq 2}(r_k;\varepsilon,f_{a_k}) & \text{if } \phi_k \geq 2 \end{cases}$$

where $p_{\phi_k=1}$ defines the syntactic coherence when the target phrase only contains one word ($\phi_k = 1$) and $p_{\phi_k\geq 2}$ defines the syntactic coherence of a target phrase composed of multiple words ($\phi_k \geq 2$). We define $p_{\phi_k=1}$ as follows:

$$p_{\phi_k=1}(r_k;\varepsilon,f_{a_k}) \propto p_n(\phi_k = 1;\varepsilon,f_{a_k})$$

where the coherence of the target phrase (word) v_k is defined to be proportional to the probability of target phrase length $\phi_k = 1$ given the source word f_{a_k}. The intuition behind this model is that the syntactic coherence is strong iff the probability of the source f_{a_k} fertility $\phi_k = 1$ is high.

For $p_{\phi_k\geq 2}$, which measures the syntactic coherence of a target phrase consisting of more than two words, we use the dependency label r_k between words $v_k[1]$ and $v_k[\phi_k]$ to indicate the level of coherence. A distribution over the values $r_k \in \mathcal{R} = \{\text{SBJ}, \text{ADJ}, \cdots\}$ ($\mathcal{R}$ is the set of dependency types for a specific language) is maintained as a table for each source word associated with all the possible lengths $\phi \in \{2, \cdots, N\}$) of the target phrase it can generate, e.g. we set $N = 4$ for ZH–EN alignment and $N = 2$ for EN–ZH alignment in our experiments.

Given a target phrase v_k containing $\phi_k(\phi_k \geq 2)$ words, it is possible that there are no dependencies between the first word $v_k[1]$ and the last word $v_k[\phi_k]$. To account for this fact, we introduce a indicator function φ as in below:

$$\varphi(v_k[1],\phi_k) = \begin{cases} 1 & \text{if } v_k[1] \text{ and } v_k[\phi_k] \text{have} \\ & \text{syntactic dependencies} \\ 0 & \text{otherwise} \end{cases}$$

We can thereafter introduce a distribution $p_\varphi(\varphi)$, where $p_\varphi(\varphi = 0) = \zeta$ $(0 \leq \zeta \leq 1)$ and $p_\varphi(\varphi = 0) = 1-\zeta$, with ζ indicating how likely it is that the first and final words in a target phrase do not have any syntactic dependencies. We can set ζ to a small number to favour target phrases satisfying the syntactic constraints and to a larger number otherwise. The introduction of this variable enables us to tune the model towards our different end goals. We can now define $p_{\phi_k\geq 2}$ as:

$$p_{\phi_k\geq 2}(r_k;\varepsilon,f_{a_k}) = p(r_k|\varphi;\varepsilon,f_{a_k})p_\varphi(\varphi)$$

where we insist that $p(r_k|\varphi;\varepsilon,f_{a_k}) = 1$ if $\varphi = 0$ (the first and last words in the target phrase do not have syntactic dependencies) to reflect the fact that in most arbitrary consecutive word sequences the first and last words do not have syntactic dependencies, and otherwise $p(r_k|\varphi;\varepsilon,f_{a_k})$ if $\varphi = 1$ (the first and last words in the target phrase have syntactic dependencies).

3.3 Parameter Estimation

The Forward-Backward Algorithm (Baum, 1972), a version of the EM algorithm (Dempster et al., 1977), is specifically designed for unsupervised parameter estimation of HMM models. The Forward statistic $\alpha_j(i,\phi,\varepsilon)$ in our model can be calculated recursively over the trellis as follows:

$$\alpha_j(i,\phi,\varepsilon) = \{\sum_{i',\phi',\varepsilon'} \alpha_{j-\phi}(i',\phi',\varepsilon')p_a(i|i',\varepsilon;I)\}$$
$$p_n(\phi;\varepsilon,f_i)\eta p_{t_1}(e_{j-\phi+1}|\varepsilon,f_i)$$
$$\prod_{j'=j-\phi+2}^{j} p_{t_2}(e_{j'}|e_{j'-1},\varepsilon,f_i)p_r(r_k;\varepsilon,f_i,\phi)$$

which sums up the probabilities of every path that could lead to the cell $\langle j,i,\phi\rangle$. Note that the syntactic coherence term $p_r(r_k;\varepsilon,f_i,\phi)$ can efficiently be added into the Forward procedure. Similarly, the Backward statistic $\beta_j(i,\phi,\varepsilon)$ is calculated over the trellis as below:

$$\beta_j(i,\phi,\varepsilon) = \sum_{i',\phi',\varepsilon'} \beta_{j+\phi'}(i',\phi',\varepsilon')p_a(i'|i,h';I)$$
$$p_n(\phi';\varepsilon',f_{i'})\eta p_{t_1}(e_{j+1}|\varepsilon',f_{i'})$$
$$\prod_{j'=j+2}^{j+\phi'} p_{t_2}(e_{j'}|e_{j'-1},\varepsilon',f_{i'})p_r(r_k;\varepsilon',f_{i'},\phi')$$

Note also that the syntactic coherence term $p_r(r_k;\varepsilon',f_{i'},\phi')$ can be integrated into the Backward procedure efficiently.

Posterior probability can be calculated based on the Forward and Backward probabilities.

3.4 EM Parameter Updates

The Expectation step accumulates fractional counts using the posterior probabilities for each parameter during the Forward-Backward passes, and the Maximisation step normalises the counts in order to generate updated parameters.

The E-step for the syntactic coherence model proceeds as follows:

$$c(r';f,\phi') = \sum_{(\mathbf{f},\mathbf{e})\in\mathbf{T}} \sum_{i,j,\phi,f_i=f} \gamma_j(i,\phi,\varepsilon=1) \\ \delta(\phi,\phi')\delta(\varphi_j(e,\phi),r')$$

where $\gamma_j(i,\phi,\varepsilon)$ is the posterior probability that a target phrase $t^j_{j-\phi+1}$ is aligned to source word f_i, and $\varphi_j(e,\phi)$ is the syntactic dependency label between $e_{j-\phi+1}$ and e_j. The M-step performs normalisation, as below:

$$p_r(r';f,\phi') = \frac{c(r';f,\phi')}{\sum_r c(r;f,\phi')}$$

Other component parameters can be estimated in a similar manner.

4 Experimental Setup

4.1 Data

We built the baseline word alignment and Phrase-Based SMT (PB-SMT) systems using existing open-source toolkits for the purposes of fair comparison. A collection of GALE data (LDC2006E26) consisting of 103K (2.9 million English running words) sentence pairs was firstly used as a proof of concept ("small"), and FBIS data containing 238K sentence pairs (8 million English running words) was added to construct a "medium" scale experiment. To investigate the intrinsic quality of the alignment, a collection of parallel sentences (12K sentence pairs) for which we have manually annotated word alignment was added to both "small" and "medium" scale experiments. Multiple-Translation Chinese Part 1 (MTC1) from LDC was used for Minimum Error-Rate Training (MERT) (Och, 2003), and MTC2, 3 and 4 were used as development test sets. Finally the test set from NIST 2006 evaluation campaign was used as the final test set.

The Chinese data was segmented using the LDC word segmenter. The maximum-entropy-based POS tagger MXPOST (Ratnaparkhi, 1996) was used to tag both English and Chinese texts. The syntactic dependencies for both English and Chinese were obtained using the state-of-the-art Maltparser dependency parser, which achieved 84% and 88% labelled attachment scores for Chinese and English respectively.

4.2 Word Alignment

The GIZA++ (Och and Ney, 2003) implementation of IBM Model 4 (Brown et al., 1993) is used as the baseline for word alignment. Model 4 is incrementally trained by performing 5 iterations of Model 1, 5 iterations of HMM, 3 iterations of Model 3, and 3 iterations of Model 4. We compared our model against the MTTK (Deng and Byrne, 2006) implementation of the HMM word-to-phrase alignment model. The model training includes 10 iterations of Model 1, 5 iterations of Model 2, 5 iterations of HMM word-to-word alignment, 20 iterations (5 iterations respectively for phrase lengths 2, 3 and 4 with unigram translation probability, and phrase length 4 with bigram translation probability) of HMM word-to-phrase alignment for ZH–EN alignment and 5 iterations (5 iterations for phrase length 2 with uniform translation probability) of HMM word-to-phrase alignment for EN–ZH. This configuration is empirically established as the best for Chinese–English word alignment. To allow for a fair comparison between IBM Model 4 and HMM word-to-phrase alignment models, we also restrict the maximum fertility in IBM model 4 to 4 for ZH–EN and 2 for EN–ZH (the default is 9 in GIZA++ for both ZH–EN and EN–ZH). "grow-diag-final" heuristic described in (Koehn et al., 2003) is used to derive the refined alignment from bidirectional alignments.

4.3 MT system

The baseline in our experiments is a standard log-linear PB-SMT system. With the word alignment obtained using the method described in

section 4.2, we perform phrase-extraction using heuristics described in (Koehn et al., 2003), Minimum Error-Rate Training (MERT) (Och, 2003) optimising the BLEU metric, a 5-gram language model with Kneser-Ney smoothing (Kneser and Ney, 1995) trained with SRILM (Stolcke, 2002) on the English side of the training data, and MOSES (Koehn et al., 2007) for decoding. A Hiero-style decoder Joshua (Li et al., 2009) is also used in our experiments. All significance tests are performed using approximate randomisation (Noreen, 1989) at $p = 0.05$.

5 Experimental Results

5.1 Alignment Model Tuning

In order to find the value of ζ in the SSH model that yields the best MT performance, we used three development test sets using a PB-SMT system trained on the small data condition. Figure 1 shows the results on each development test set using different configurations of the alignment models. For each system, we obtain the mean of the BLEU scores (Papineni et al., 2002) on the three development test sets, and derive the optimal value for ζ of 0.4, which we use hereafter for final testing. It is worth mentioning that while IBM model 4 (M4) outperforms other models including the HMM word-to-word (H) and word-to-phrase (SH) alignment model in our current setup, using the default IBM model 4 setting (maximum fertility 9) yields an inferior performance (as much as 8.5% relative) compared to other models.

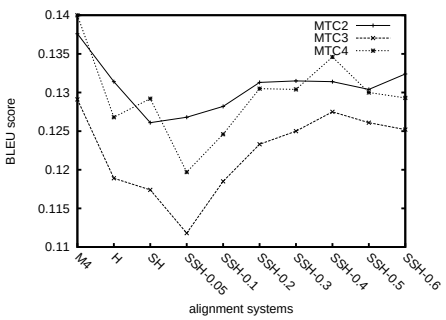


Figure 1: BLEU score on development test set using PB-SMT system

	PB-SMT		Hiero	
	small	medium	small	medium
H	0.1440	0.2591	0.1373	0.2595
SH	0.1418	0.2517	0.1372	0.2609
SSH	0.1464	0.2518	0.1356	0.2624
M4	0.1566	0.2627	0.1486	0.2660

Table 1: Performance of PB-SMT using different alignment models on NIST06 test set

5.2 Translation Results

Table 1 shows the performance of PB-SMT and Hiero systems using a small amount of data for alignment model training on the NIST06 test set. For the PB-SMT system trained on the small data set, using SSH word alignment leads to a 3.24% relative improvement over SH, which is statistically significant. SSH also leads to a slight gain over the HMM word-to-word alignment model (H). However, when the PB-SMT system is trained on larger data sets, there are no significant differences between SH and SSH. Additionally, both SH and SSH models underperform H on the medium data condition, indicating that the performance of the alignment model tuned on the PB-SMT system with small training data does not carry over to PB-SMT systems with larger training data (cf. Figure 1). IBM model 4 demonstrates stronger performance over other models for both small and medium data conditions.

For the Hiero system trained on a small data set, no significant differences are observed between SSH, SH and H. On a larger training set, we observe that SSH alignment leads to better performance compared to SH. Both SH and SSH alignments achieved higher translation quality than H. Note that while IBM model 4 outperforms other models on a small data condition, the difference between IBM model 4 and SSH is not statistically significant on a medium data condition. It is also worth pointing out that the SSH model yields significant improvement over IBM model 4 with the default fertility setting, indicating that varying the fertility limit in IBM model 4 has a significant impact on translation quality.

In summary, the SSH model which incorporates syntactic dependencies into the SH model achieves consistently better performance than

	ZH–EN		EN–ZH	
	P	R	P	R
H	0.5306	0.3752	0.5282	0.3014
SH	0.5378	0.3802	0.5523	0.3151
SSH	0.5384	0.3807	0.5619	0.3206
M4	0.5638	0.3986	0.5988	0.3416

Table 2: Intrinsic evaluation of the alignment using different alignment models

SH in both PB-SMT and Hiero systems under both small and large data conditions. For a PB-SMT system trained on the small data set, the SSH model leads to significant gains over the baseline SH model. The results also entail an observation concerning the suitability of different alignment models for different types of SMT systems; trained on a large data set, our SSH alignment model is more suitable to a Hiero-style system than a PB-SMT system, as evidenced by a lower performance compared to IBM model 4 using a PB-SMT system, and a comparable performance compared to IBM model 4 using a Hiero system.

5.3 Intrinsic Evaluation

In order to further investigate the intrinsic quality of the word alignment, we compute the Precision (P), Recall (R) and F-score (F) of the alignments obtained using different alignment models. As the models investigated here are asymmetric models, we conducted intrinsic evaluation for both alignment directions, i.e. ZH–EN word alignment where one Chinese word can be aligned to multiple English words, and EN–ZH word alignment where one English word can be aligned to multiple Chinese words.

Table 2 shows the results of the intrinsic evaluation of ZH–EN and EN–ZH word alignment on a small data set (results on the medium data set follow the same trend but are left out due to space limitations). Note that the P and R are all quite low, demonstrating the difficulty of Chinese–English word alignment in the news domain. For the ZH–EN direction, using the SSH model does not lead to significant gains over SH in P or R. For the EN–ZH direction, the SSH model leads to a 1.74% relative improvement in P, and a 1.75% relative improvement in R over the SH model. Both SH and SSH lead to gains over H for both ZH–EN and EN–ZH directions, while gains in the EN–ZH direction appear to be more pronounced. IBM model 4 achieves significantly higher P over other models while the gap in R is narrow.

Relating Table 2 to Table 1, we observe that the HMM word-to-word alignment model (H) can still achieve good MT performance despite the lower P and R compared to other models. This provides additional support to previous findings (Fraser and Marcu, 2007b) that the intrinsic quality of word alignment does not necessarily correlate with the performance of the resulted MT system.

5.4 Alignment Characteristics

In order to further understand the characteristics of the alignment that each model produces, we investigated several statistics of the alignment results which can hopefully reveal the capabilities and limitations of each model.

5.4.1 Pairwise Comparison

Given the asymmetric property of these alignment models, we can evaluate the quality of the links for each word and compare the alignment links across different models. For example, in ZH–EN word alignment, we can compute the links for each Chinese word and compare those links across different models. Additionally, we can compute the pairwise agreement in aligning each Chinese word for any two alignment models. Similarly, we can compute the pairwise agreement in aligning each English word in the EN–ZH alignment direction.

For ZH–EN word alignment, we observe that the SH and SSH models reach a 85.94% agreement, which is not surprising given the fact that SSH is a syntactic extension over SH, while IBM model 4 and SSH reach the smallest agreement (only 65.09%). We also observe that there is a higher agreement between SSH and H (76.64%) than IBM model 4 and H (69.58%). This can be attributed to the fact that SSH is still a form of HMM model while IBM model 4 is not. A similar trend is observed for EN–ZH word alignment.

	ZH–EN				EN–ZH			
	1-to-0	1-to-1	1-to-n		1-to-0	1-to-1	1-to-n	
			con.	non-con.			con.	non-con.
HMM	0.3774	0.4693	0.0709	0.0824	0.4438	0.4243	0.0648	0.0671
SH	0.3533	0.4898	0.0843	0.0726	0.4095	0.4597	0.0491	0.0817
SSH	0.3613	0.5092	0.0624	0.0671	0.3990	0.4835	0.0302	0.0872
M4	0.2666	0.5561	0.0985	0.0788	0.3967	0.4850	0.0592	0.0591

Table 3: Alignment types using different alignment models

5.4.2 Alignment Types

Again, by taking advantage of the asymmetric property of these alignment models, we can compute different types of alignment. For both ZH–EN (EN–ZH) alignment, we divide the links for each Chinese (English) word into 1-to-0 where each Chinese (English) word is aligned to the empty word "NULL" in English (Chinese), 1-to-1 where each Chinese (English) word is aligned to only one word in English (Chinese), and 1-to-n where each Chinese (English) word is aligned to n ($n \geq 2$) words in English (Chinese). For 1-to-n links, depending on whether the n words are consecutive, we have consecutive (con.) and non-consecutive (non-con.) 1-to-n links.

Table 3 shows the alignment types in the medium data track. We can observe that for ZH–EN word alignment, both SH and SSH produce far more 1-to-0 links than Model 4. It can also be seen that Model 4 tends to produce more consecutive 1-to-n links than non-consecutive 1-to-n links. On the other hand, the SSH model tends to produce more non-consecutive 1-to-n links than consecutive ones. Compared to SH, SSH tends to produce more 1-to-1 links than 1-to-n links, indicating that adding syntactic dependency constraints biases the model towards only producing 1-to-n links when the n words follow coherence constraint, i.e. the first and last word in the chunk have syntactic dependencies. For example, among the 6.24% consecutive ZH–EN 1-to-n links produced by SSH, 43.22% of them follow the coherence constraint compared to just 39.89% in SH. These properties can have significant implications for the performance of our MT systems given that we use the grow-diag-final heuristics to derive the symmetrised word alignment based on bidirectional asymmetric word alignments.

6 Conclusions and Future Work

In this paper, we extended the HMM word-to-phrase word alignment model to handle syntactic dependencies. We found that our model was consistently better than that without syntactic dependencies according to both intrinsic and extrinsic evaluation. Our model is shown to be beneficial to PB-SMT under a small data condition and to a Hiero-style system under a larger data condition.

As to future work, we firstly plan to investigate the impact of parsing quality on our model, and the use of different heuristics to combine word alignments. Secondly, the syntactic coherence model itself is very simple, in that it only covers the syntactic dependency between the first and last word in a phrase. Accordingly, we intend to extend this model to cover more sophisticated syntactic relations within the phrase. Furthermore, given that we can construct different MT systems using different word alignments, multiple system combination can be conducted to avail of the advantages of different systems. We also plan to compare our model with other alignment models, e.g. (Fraser and Marcu, 2007a), and test this approach on more data and on different language pairs and translation directions.

Acknowledgements

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. Part of the work was carried out at Cambridge University Engineering Department with Dr. William Byrne. The authors would also like to thank the anonymous reviewers for their insightful comments.

References

Baum, Leonard E. 1972. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3:1–8.

Blunsom, Phil, Trevor Cohn, Chris Dyer, and Miles Osborne. 2009. A gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of ACL-IJCNLP 2009*, pages 782–790, Singapore.

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.

Cherry, Colin and Dekang Lin. 2006. Soft syntactic constraints for word alignment through discriminative training. In *Proceedings of the COLING-ACL 2006*, pages 105–112, Sydney, Australia.

Dempster, Arthur, Nan Laird, and Donald Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.

DeNero, John and Dan Klein. 2008. The complexity of phrase alignment problems. In *Proceedings of ACL-08: HLT, Short Papers*, pages 25–28, Columbus, OH.

Deng, Yonggang and William Byrne. 2006. MTTK: An alignment toolkit for Statistical Machine Translation. In *Proceedings of HLT-NAACL 2006*, pages 265–268, New York City, NY.

Deng, Yonggang and William Byrne. 2008. HMM word and phrase alignment for Statistical Machine Translation. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(3):494–507.

Fox, Heidi. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of the EMNLP 2002*, pages 304–3111, Philadelphia, PA, July.

Fraser, Alexander and Daniel Marcu. 2007a. Getting the structure right for word alignment: LEAF. In *Proceedings of EMNLP-CoNLL 2007*, pages 51–60, Prague, Czech Republic.

Fraser, Alexander and Daniel Marcu. 2007b. Measuring word alignment quality for Statistical Machine Translation. *Computational Linguistics*, 33(3):293–303.

Kneser, Reinhard and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE ICASSP*, volume 1, pages 181–184, Detroit, MI.

Koehn, Philipp, Franz Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of HLT-NAACL 2003*, pages 48–54, Edmonton, AB, 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 *Proceedings of ACL 2007*, pages 177–180, Prague, Czech Republic.

Li, Zhifei, Chris Callison-Burch, Chris Dyer, Sanjeev Khudanpur, Lane Schwartz, Wren Thornton, Jonathan Weese, and Omar Zaidan. 2009. Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the WMT 2009*, pages 135–139, Athens, Greece.

Marcu, Daniel and William Wong. 2002. A Phrase-Based, joint probability model for Statistical Machine Translation. In *Proceedings of EMNLP 2002*, pages 133–139, Philadelphia, PA.

Murphy, Kevin. 2002. Hidden semi-markov models (segment models). Technical report, UC Berkeley.

Nivre, Joakim, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Ervin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Noreen, Eric W. 1989. *Computer-Intensive Methods for Testing Hypotheses: An Introduction*. Wiley-Interscience, New York, NY.

Och, Franz and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Och, Franz. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of ACL 2003*, pages 160–167, Sapporo, Japan.

Ostendorf, Mari, Vassilios V. Digalakis, and Owen A. Kimball. 1996. From HMMs to segment models: A unified view of stochastic modeling for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 4(5):360–378.

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*, pages 311–318, Philadelphia, PA.

Ratnaparkhi, Adwait. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of EMNLP 1996*, pages 133–142, Somerset, NJ.

Stolcke, Andreas. 2002. SRILM – An extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904, Denver, CO.

Vogel, Stefan, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of ACL 1996*, pages 836–841, Copenhagen, Denmark.

New Parameterizations and Features for PSCFG-Based Machine Translation

Andreas Zollmann Stephan Vogel
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
{zollmann,vogel}@cs.cmu.edu

Abstract

We propose several improvements to the hierarchical phrase-based MT model of Chiang (2005) and its syntax-based extension by Zollmann and Venugopal (2006). We add a source-span variance model that, for each rule utilized in a probabilistic synchronous context-free grammar (PSCFG) derivation, gives a confidence estimate in the rule based on the number of source words spanned by the rule and its substituted child rules, with the distributions of these source span sizes estimated during training time.

We further propose different methods of combining hierarchical and syntax-based PSCFG models, by merging the grammars as well as by interpolating the translation models.

Finally, we compare syntax-augmented MT, which extracts rules based on target-side syntax, to a corresponding variant based on source-side syntax, and experiment with a model extension that jointly takes source and target syntax into account.

1 Introduction

The Probabilistic Synchronous Context Free Grammar (PSCFG) formalism suggests an intuitive approach to model the long-distance and lexically sensitive reordering phenomena that often occur across language pairs considered for statistical machine translation. As in monolingual parsing, nonterminal symbols in translation rules are used to generalize beyond purely lexical operations. *Labels* on these nonterminal symbols are often used to enforce syntactic constraints in the generation of bilingual sentences and imply conditional independence assumptions in the statistical translation model. Several techniques have been recently proposed to automatically identify and estimate parameters for PSCFGs (or related synchronous grammars) from parallel corpora (Galley et al., 2004; Chiang, 2005; Zollmann and Venugopal, 2006; Liu et al., 2006; Marcu et al., 2006).

In this work, we propose several improvements to the hierarchical phrase-based MT model of Chiang (2005) and its syntax-based extension by Zollmann and Venugopal (2006). We add a source span variance model that, for each rule utilized in a probabilistic synchronous context-free grammar (PSCFG) derivation, gives a confidence estimate in the rule based on the number of source words spanned by the rule and its substituted child rules, with the distributions of these source span sizes estimated during training (i.e., rule extraction) time.

We further propose different methods of combining hierarchical and syntax-based PSCFG models, by merging the grammars as well as by interpolating the translation models.

Finally, we compare syntax-augmented MT, which extracts rules based on target-side syntax, to a corresponding variant based on source-side syntax, and experiment with a model extension based on source *and* target syntax.

We evaluate the different models on the NIST large resource Chinese-to-English translation task.

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 110–117,
COLING 2010, Beijing, August 2010.

2 Related work

Chiang et al. (2008) introduce *structural distortion features* into a hierarchical phrase-based model, aimed at modeling nonterminal reordering given source span length, by estimating for each possible source span length ℓ a Bernoulli distribution $p(R|\ell)$ where R takes value one if reordering takes place and zero otherwise. Maximum-likelihood estimation of the distribution amounts to simply counting the relative frequency of nonterminal reorderings over all extracted rule instances that incurred a substitution of span length ℓ. In a more fine-grained approach they add a separate binary feature $\langle R, \ell \rangle$ for each combination of reordering truth value R and span length ℓ (where all $\ell \geq 10$ are merged into a single value), and then tune the feature weights discriminatively on a development set. Our approach differs from Chiang et al. (2008) in that we estimate one source span length distribution for each substitution site of each grammar rule, resulting in unique distributions for each rule, estimated from all instances of the rule in the training data. This enables our model to condition reordering range on the individual rules used in a derivation, and even allows to distinguish between two rules r_1 and r_2 that both reorder arguments with identical mean span lengths ℓ, but where the span lengths encountered in extracted instances of r_1 are all close to ℓ, whereas span length instances for r_2 vary widely.

Chen and Eisele (2010) propose a hypbrid approach between hierarchical phrase based MT and a rule based MT system, reporting improvement over each individual model on an English-to-German translation task. Essentially, the rule based system is converted to a single-nonterminal PSCFG, and hence can be combined with the hierarchical model, another single-nonterminal PSCFG, by taking the union of the rule sets and augmenting the feature vectors, adding zero-values for rules that only exist in one of the two grammars. We face the challenge of combining the single-nonterminal hierarchical grammar with a multi-nonterminal syntax-augmented grammar. Thus one hierarchical rule typically corresponds to many syntax-augmented rules. The SAMT system used by Zollmann et al. (2008) adds hierarchical rules separately to the syntax-augmented grammar, resulting in a backbone grammar of well-estimated hierarchical rules supporting the sparser syntactic rules. They allow the model preference between hierarchical and syntax rules to be learned from development data by adding an indicator feature to all rules, which is one for hierarchical rules and zero for syntax rules. However, no empirical comparison is given between the purely syntax-augmented and the hybrid grammar. We aim to fill this gap by experimenting with both models, and further refine the hybrid approach by adding interpolated probability models to the syntax rules.

Chiang (2010) augments a hierarchical phrase-based MT model with binary syntax features representing the source and target syntactic constituents of a given rule's instantiations during training, thus taking source and target syntax into account while avoiding the data-sparseness and decoding-complexity problems of multi-nonterminal PSCFG models. In our approach, the source- and target-side syntax directly determines the grammar, resulting in a nonterminal set derived from the labels underlying the source- and target-language treebanks.

3 PSCFG-based translation

Given a source language sentence $\mathbf{f}$, statistical machine translation defines the translation task as selecting the most likely target translation $\mathbf{e}$ under a model $P(\mathbf{e}|\mathbf{f})$, i.e.:

$$\hat{\mathbf{e}}(\mathbf{f}) = \arg\max_{\mathbf{e}} P(\mathbf{e}|\mathbf{f}) = \arg\max_{\mathbf{e}} \sum_{i=1}^{m} h_i(\mathbf{e}, \mathbf{f})\lambda_i$$

where the $\arg\max$ operation denotes a search through a structured space of translation outputs in the target language, $h_i(\mathbf{e}, \mathbf{f})$ are bilingual features of $\mathbf{e}$ and $\mathbf{f}$ and monolingual features of $\mathbf{e}$, and weights λ_i are typically trained discriminatively to maximize translation quality (based on automatic metrics) on held out data, e.g., using minimum-error-rate training (MERT) (Och, 2003).

In PSCFG-based systems, the search space is structured by automatically extracted rules that model both translation and re-ordering operations.

Most large scale systems approximate the search above by simply searching for the most likely derivation of rules, rather than searching for the most likely translated output. There are efficient algorithms to perform this search (Kasami, 1965; Chappelier and Rajman, 1998) that have been extended to efficiently integrate n-gram language model features (Chiang, 2007; Venugopal et al., 2007; Huang and Chiang, 2007; Zollmann et al., 2008; Petrov et al., 2008).

In this work we experiment with PSCFGs that have been automatically learned from word-aligned parallel corpora. PSCFGs are defined by a source terminal set (source vocabulary) $\mathcal{T}_S$, a target terminal set (target vocabulary) $\mathcal{T}_T$, a shared nonterminal set $\mathcal{N}$ and rules of the form: $X \rightarrow \langle \gamma, \alpha, w \rangle$ where

- $X \in \mathcal{N}$ is a labeled nonterminal referred to as the left-hand-side of the rule.
- $\gamma \in (\mathcal{N} \cup \mathcal{T}_S)*$ is the source side of the rule.
- $\alpha \in (\mathcal{N} \cup \mathcal{T}_T)*$ is the target side of the rule.
- $w \in [0, \infty)$ is a non-negative real-valued weight assigned to the rule; in our model, w is the exponential function of the inner product of features h and weights λ.

3.1 Hierarchical phrase-based MT

Building upon the success of phrase-based methods, Chiang (2005) presents a PSCFG model of translation that uses the bilingual phrase pairs of phrase-based MT as starting point to learn hierarchical rules. For each training sentence pair's set of extracted phrase pairs, the set of induced PSCFG rules can be generated as follows: First, each phrase pair is assigned a generic X-nonterminal as left-hand-side, making it an *initial rule*. We can now recursively generalize each already obtained rule (initial or including nonterminals)

$$N \rightarrow f_1 \dots f_m / e_1 \dots e_n$$

for which there is an *initial* rule

$$M \rightarrow f_i \dots f_u / e_j \dots e_v$$

where $1 \leq i < u \leq m$ and $1 \leq j < v \leq n$, to obtain a new rule

$$N \rightarrow f_1^{i-1} X_k f_{u+1}^m / e_1^{j-1} X_k e_{v+1}^n$$

where e.g. f_1^{i-1} is short-hand for $f_1 \dots f_{i-1}$, and where k is an index for the nonterminal X that indicates the one-to-one correspondence between the new X tokens on the two sides (it is not in the space of word indices like i, j, u, v, m, n). The recursive form of this generalization operation allows the generation of rules with multiple nonterminal pairs.

Chiang (2005) uses features analogous to the ones used in phrase-based translation: a language model neg-log probability, a 'rule given source-side' neg-log-probability, a 'rule given target-side' neg-log-probability, source- and target conditioned 'lexical' neg-log-probabilities based on word-to-word co-occurrences (Koehn et al., 2003), as well as rule, target word, and glue operation counters. We follow Venugopal and Zollmann (2009) to further add a rareness penalty,

$$1/\operatorname{count}(r)$$

where $\operatorname{count}(r)$ is the occurrence count of rule r in the training corpus, allowing the system to learn penalization of low-frequency rules, as well as three indicator features firing if the rule has one, two unswapped, and two swapped nonterminal pairs, respectively.[1]

3.2 Syntax Augmented MT

Syntax Augmented MT (SAMT) (Zollmann and Venugopal, 2006) extends Chiang (2005) to include nonterminal symbols from target language phrase structure parse trees. Each target sentence in the training corpus is parsed with a stochastic parser to produce constituent labels for target spans. Phrase pairs (extracted from a particular sentence pair) are assigned left-hand-side nonterminal symbols based on the target side parse tree constituent spans.

Phrase pairs whose target side corresponds to a constituent span are assigned that constituent's label as their left-hand-side nonterminal. If the target side of the phrase pair is not spanned by a single constituent in the corresponding parse tree, we use the labels of subsuming, subsumed, and neighboring parse tree constituents to assign

[1] Penalization or reward of purely-lexical rules can be indirectly learned by trading off these features with the rule counter feature.

an extended label of the form $C_1 + C_2$, C_1/C_2, or $C_2\backslash C_1$ (the latter two being motivated from the operations in combinatory categorial grammar (CCG) (Steedman, 2000)), indicating that the phrase pair's target side spans two adjacent syntactic categories (e.g., *she went: NP+VB*), a partial syntactic category C_1 missing a C_2 at the right (e.g., *the great: NP/NN*), or a partial C_1 missing a C_2 at the left (e.g., *great wall: DT\NP*), respectively. The label assignment is attempted in the order just described, i.e., assembling labels based on '+' concatenation of two subsumed constituents is preferred, as smaller constituents tend to be more accurately labeled. If no label is assignable by either of these three methods, a default label 'FAIL' is assigned.

In addition to the features used in hierarchical phrase-based MT, SAMT introduces a relative-frequency estimated probability of the rule given its left-hand-side nonterminal.

4 Modeling Source Span Length of PSCFG Rule Substitution Sites

Extracting a rule with k right-hand-side nonterminal pairs, i.e., substitution sites, (from now on called *order-k rule*) by the method described in Section 3 involves $k + 1$ phrase pairs: one phrase pair used as initial rule and k phrase pairs that are sub phrase pairs of the first and replaced by nonterminal pairs. Conversely, during translation, applying this rule amounts to combining k hypotheses from k different chart cells, each represented by a source span and a nonterminal, to form a new hypothesis and file it into a chart cell. Intuitively, we want the source span lengths of these $k + 1$ chart cells to be close to the source side lengths of the $k+1$ phrase pairs from the training corpus that were involved in extracting the rule. Of course, each rule generally was extracted from multiple training corpus locations, with different involved phrase pairs of different lengths. We therefore model $k + 1$ source span length distributions for each order-k rule in the grammar.

Ignoring the discreteness of source span length for the sake of easier estimation, we assume the distribution to be log-normal. This is motivated by the fact that source span length is positive and that we expect its deviation between instances of the same rule to be greater for long phrase pairs than for short ones.

We can now add $\hat{k} + 1$ features to the translation framework, where $\hat{k}$ is the maximum number of PSCFG rule nonterminal pairs, in our case two. Each feature is computed during translation time. Ideally, it should represent the probability of the hypothesized rule given the respective chart cell span length. However, as each competing rule underlies a different distribution, this would require a Bayesian setting, in which priors over distributions are specified. In this preliminary work we take a simpler approach: Based on the rule's span distribution, we compute the probability that a span length no likelier than the one encountered was generated from the distribution. This probability thus yields a confidence estimate for the rule. More formally, let μ be the mean and σ the standard deviation of the logarithm of the span length random variable X concerned, and let x be the span length encountered during decoding. Then the computed confidence estimate is given by

$$P\left(|\ln(X) - \mu| \geq |\ln(x) - \mu|\right) = 2 * Z\left(-(|\ln(x) - \mu|)/\sigma\right)$$

where Z is the cumulative density function of the normal distribution with mean zero and variance one.

The confidence estimate is one if the encountered span length is equal to the mean of the distribution, and decreases as the encountered span length deviates further from the mean. The severity of that decline is determined by the distribution variance: the higher the variance, the less a deviation from the mean is penalized.

Mean and variance of log source span length are sufficient statistics of the log-normal distribution. As we extract rules in a distributed fashion, we use a straightforward parallelization of the online algorithm of Welford (1962) and its improvement by West (1979) to compute the sample variance over all instances of a rule.

5 Merging a Hierarchical and a Syntax-Based Model

While syntax-based grammars allow for more refined statistical models and guide the search by constraining substitution possibilitites in a grammar derivation, grammar sizes tend to be much greater than for hierarchical grammars. Therefore the average occurrence count of a syntax rule is much lower than that of a hierarchical rule, and thus estimated probabilitites are less reliable.

We propose to augment the syntax-based "rule given source side" and "rule given target side" distributions by hierarchical counterparts obtained by marginalizing over the left-hand-side and right-hand-side rule nonterminals. For example, the hierarchical equivalent of the "rule given source side" probability is obtained by summing occurrence counts over all rules that have the same source and target terminals and substitution positions but possibly differ in the left- and/or right-hand side nonterminal labels, divided by the sum of occurrence counts of all rules that have the same source side terminals and source side substitution positions. Similarly, an alternative rareness penalty based on the combined frequency of all rules with the same terminals and substitution positions is obtained.

Using these syntax and hierarchical features side by side amounts to interpolation of the respective probability models in log-space, with minimum-error-rate training (MERT) determining the optimal interpolation coefficient. We also add respective models interpolated with coefficient .5 in probability-space as additional features to the system.

We further experiment with adding hierarchical rules separately to the syntax-augmented grammar, as proposed in Zollmann et al. (2008), with the respective syntax-specific features set to zero. A 'hierarchical-indicator' feature is added to all rules, which is one for hierarchical rules and zero for syntax rules, allowing the joint model to trade of hierarchical against syntactic rules. During translation, the hierarchical and syntax worlds are bridged by glue rules, which allow monotonic concatenation of hierarchical and syntactic partial sentence hypotheses. We separate the glue feature used in hierarchical and syntax-augmented translation into a glue feature that only fires when a hierarchical rule is glued, and a distinct glue feature firing when gluing a syntax-augmented rule.

6 Extension of SAMT to a bilingually parsed corpus

Syntax-based MT models have been proposed both based on target-side syntactic annotations (Galley et al., 2004; Zollmann and Venugopal, 2006) as well source-side annotations (Liu et al., 2006). Syntactic annotations for both source *and* target language are available for popular language pairs such as Chinese-English. In this case, our grammar extraction procedure can be easily extended to impose both source and target constraints on the eligible substitutions simultaneously.

Let N_f be the nonterminal label that would be assigned to a given initial rule when utilizing the source-side parse tree, and N_e the assigned label according to the target-side parse. Then our bilingual model assigns '$N_f + N_e$' to the initial rule. The extraction of complex rules proceeds as before. The number of nonterminals in this model, based on a source-model label set of size s and a target label set of size t, is thus given by st.

7 Experiments

We evaluate our approaches by comparing translation quality according to the IBM-BLEU (Papineni et al., 2002) metric on the NIST Chinese-to-English translation task using MT04 as development set to train the model parameters λ, and MT05, MT06 and MT08 as test sets.

We perform PSCFG rule extraction and decoding using the open-source "SAMT" system (Venugopal and Zollmann, 2009), using the provided implementations for the hierarchical and syntax-augmented grammars. For all systems, we use the bottom-up chart parsing decoder implemented in the SAMT toolkit with a reordering limit of 15 source words, and correspondingly extract rules from initial phrase pairs of maximum source length 15. All rules have at most two nonterminal symbols, which must be non-consecutive on the source side, and rules must contain at least

one source-side terminal symbol.

For parameter tuning, we use the L_0-regularized minimum-error-rate training tool provided by the SAMT toolkit.

The parallel training data comprises of 9.6M sentence pairs (206M Chinese Words, 228M English words). The source and target language parses for the syntax-augmented grammar were generated by the Stanford parser (Klein and Manning, 2003).

The results are given in Table 1. The source span models (indicated by +span) achieve small test set improvements of 0.15 BLEU points on average for the hierarchical and 0.26 BLEU points for the syntax-augmented system, but these are not statistically significant.

Augmenting a syntax-augmented grammar with hierarchical features ("Syntax+hiermodels") results in average test set improvements of 0.5 BLEU points. These improvements are not statistically significant either, but persist across all three test sets. This demonstrates the benefit of more reliable feature estimation. Further augmenting the hierarchical rules to the grammar ("Syntax+hiermodels+hierrules") does not yield additional improvements.

The use of bilingual syntactic parses ('Syntax/src&tgt') turns out detrimental to translation quality. We assume this is due to the huge number of nonterminals in these grammars and the great amount of badly-estimated low-occurrence-count rules. Perhaps merging this grammar with a regular syntax-augmented grammar could yield better results.

We also experimented with a source-parse based model ('Syntax/src'). While not being able to match translation quality of its target-based counterpart, the model still outperforms the hierarchical system on all test sets.

8 Conclusion

We proposed several improvements to the hierarchical phrase-based MT model of Chiang (2005) and its syntax-based extension by Zollmann and Venugopal (2006). We added a source span length model that, for each rule utilized in a probabilistic synchronous context-free grammar (PSCFG) derivation, gives a confidence estimate in the rule based on the number of source words spanned by the rule and its substituted child rules, resulting in small improvements for hierarchical phrase-based as well as syntax-augmented MT.

We further demonstrated the utility of combining hierarchical and syntax-based PSCFG models and grammars.

Finally, we compared syntax-augmented MT, which extracts rules based on target-side syntax, to a corresponding variant based on source-side syntax, showing that target syntax is more benefitial, and unsuccessfully experimented with a model extension that jointly takes source and target syntax into account.

Hierarchical phrase-based MT suffers from spurious ambiguity: A single translation for a given source sentence can usually be accomplished by many different PSCFG derivations. This problem is exacerbated by syntax-augmented MT with its thousands of nonterminals, and made even worse by its joint source-and-target extension. Future research should apply the work of Blunsom et al. (2008) and Blunsom and Osborne (2008), who marginalize over derivations to find the most probable translation rather than the most probable derivation, to these multi-nonterminal grammars.

All source code underlying this work is available under the GNU Lesser General Public License as part of the 'SAMT' system at:
`www.cs.cmu.edu/~zollmann/samt`

Acknowledgements

This work is in part supported by NSF under the Cluster Exploratory program (grant NSF 0844507), and in part by the US DARPA GALE program. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF or DARPA.

References

Blunsom, Phil and Miles Osborne. 2008. Probabilistic inference for machine translation. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 215–223, Morristown, NJ, USA. Association for Computational Linguistics.

	Dev (MT04)	MT05	MT06	MT08	**TestAvg**	Time
Hierarchical	38.63	36.51	33.26	25.77	**31.85**	14.3
Hier+span	39.03	36.44	33.29	26.26	**32.00**	16.7
Syntax	39.17	37.17	33.87	26.81	**32.62**	59
Syntax+hiermodels	39.61	37.74	34.30	27.30	**33.11**	68.4
Syntax+hiermodels+hierrules	39.69	37.56	34.66	26.93	**33.05**	34.6
Syntax+span+hiermodels+hierrules	39.81	38.02	34.50	27.41	**33.31**	39.6
Syntax/src+span+hiermodels+hierrules	39.62	37.25	33.99	26.44	**32.56**	20.1
Syntax/src&tgt+span+hiermodels+hierrules	39.15	36.92	33.70	26.24	**32.29**	17.5

Table 1: Translation quality in % case-insensitive IBM-BLEU (i.e., brevity penalty based on closest reference length) for different systems on Chinese-English NIST-large translation tasks. 'TestAvg' shows the average score over the three test sets. 'Time' is the average decoding time per sentence in seconds on one CPU.

Blunsom, Phil, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

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).

Chappelier, J.C. and M. Rajman. 1998. A generalized CYK algorithm for parsing stochastic CFG. In *Proceedings of Tabulation in Parsing and Deduction (TAPD)*, pages 133–137, Paris.

Chen, Yu and Andreas Eisele. 2010. Hierarchical hybrid translation between english and german. In Hansen, Viggo and Francois Yvon, editors, *Proceedings of the 14th Annual Conference of the European Association for Machine Translation*, pages 90–97. EAMT, EAMT, 5.

Chiang, David, Yuval Marton, and Philip Resnik. 2008. Online large-margin training of syntactic and structural translation features. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 224–233, Honolulu, Hawaii, October. Association for Computational Linguistics.

Chiang, David. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Chiang, David. 2007. Hierarchical phrase based translation. *Computational Linguistics*, 33(2).

Chiang, David. 2010. Learning to translate with source and target syntax. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1443–1452, Uppsala, Sweden, July. Association for Computational Linguistics.

Galley, Michael, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics Conference (HLT/NAACL)*.

Huang, Liang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Kasami, T. 1965. An efficient recognition and syntax-analysis algorithm for context-free languages. Technical report, Air Force Cambridge Research Lab.

Klein, Dan and Christoper Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Koehn, Philipp, Franz J. Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics Conference (HLT/NAACL)*.

Liu, Yang, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*.

Marcu, Daniel, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Sydney, Australia.

Och, Franz J. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Petrov, Slav, Aria Haghighi, and Dan Klein. 2008. Coarse-to-fine syntactic machine translation using language projections. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Steedman, Mark. 2000. *The Syntactic Process*. MIT Press.

Venugopal, Ashish and Andreas Zollmann. 2009. Grammar based statistical MT on Hadoop: An end-to-end toolkit for large scale PSCFG based MT. *The Prague Bulletin of Mathematical Linguistics*, 91:67–78.

Venugopal, Ashish, Andreas Zollmann, and Stephan Vogel. 2007. An efficient two-pass approach to synchronous-CFG driven statistical MT. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics Conference (HLT/NAACL)*.

Welford, B. P. 1962. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420.

West, D. H. D. 1979. Updating mean and variance estimates: an improved method. *Commun. ACM*, 22(9):532–535.

Zollmann, Andreas and Ashish Venugopal. 2006. Syntax augmented machine translation via chart parsing. In *Proceedings of the Workshop on Statistical Machine Translation, HLT/NAACL*.

Zollmann, Andreas, Ashish Venugopal, Franz J. Och, and Jay Ponte. 2008. A systematic comparison of phrase-based, hierarchical and syntax-augmented statistical MT. In *Proceedings of the Conference on Computational Linguistics (COLING)*.

Deep Syntax Language Models and Statistical Machine Translation

Yvette Graham
NCLT
Dublin City University
ygraham@computing.dcu.ie

Josef van Genabith
CNGL
Dublin City University
josef@computing.dcu.ie

Abstract

Hierarchical Models increase the reordering capabilities of MT systems by introducing non-terminal symbols to phrases that map source language (SL) words/phrases to the correct position in the target language (TL) translation. Building translations via discontiguous TL phrases increases the difficulty of language modeling, however, introducing the need for heuristic techniques such as cube pruning (Chiang, 2005), for example. An additional possibility to aid language modeling in hierarchical systems is to use a language model that models fluency of words not using their local context in the string, as in traditional language models, but instead using the deeper context of a word. In this paper, we explore the potential of deep syntax language models providing an interesting comparison with the traditional string-based language model. We include an experimental evaluation that compares the two kinds of models independently of any MT system to investigate the possible potential of integrating a deep syntax language model into Hierarchical SMT systems.

1 Introduction

In Phrase-Based Models of Machine Translation all phrases consistent with the word alignment are extracted (Koehn et al., 2003), with shorter phrases needed for high coverage of unseen data and longer phrases providing improved fluency in target language translations. Hierarchical Models (Chiang, 2007; Chiang, 2005) build on Phrase-Based Models by relaxing the constraint that phrases must be contiguous sequences of words and allow a short phrase (or phrases) nested within a longer phrase to be replaced by a non-terminal symbol forming a new hierarchical phrase. Traditional language models use the local context of words to estimate the probability of the sentence and introducing hierarchical phrases that generate discontiguous sequences of TL words increases the difficulty of computing language model probabilities during decoding and require sophisticated heuristic language modeling techniques (Chiang, 2007; Chiang, 2005).

Leaving aside heuristic language modeling for a moment, the difficulty of integrating a traditional string-based language model into the decoding process in a hierarchical system, highlights a slight incongruity between the translation model and language model in Hierarchical Models. According to the translation model, the best way to build a fluent TL translation is via discontiguous phrases, while the language model can only provide information about the fluency of contiguous sequences of words. Intuitively, a language model that models fluency between discontiguous words may be well-suited to hierarchical models. Deep syntax language models condition the probability of a word on its deep context, i.e. words linked to it via dependency relations, as opposed to preceding words in the string. During decoding in Hierarchical Models, words missing a context in the string due to being preceded by a non-terminal, might however be in a dependency relation with a word that is already present in the string and

Proceedings of SSST-4, Fourth Workshop on Syntax and Structure in Statistical Translation, pages 118–126,
COLING 2010, Beijing, August 2010.

this context could add useful information about the fluency of the hypothesis as its constructed.

In addition, using the deep context of a word provides a deeper notion of fluency than the local context provides on its own and this might be useful to improve such things as lexical choice in SMT systems. Good lexical choice is very important and the deeper context of a word, if available, may provide more meaningful information and result in better lexical choice. Integrating such a model into a Hierarchical SMT system is not straightforward, however, and we believe before embarking on this its worthwhile to evaluate the model independently of any MT system. We therefore provide an experimental evaluation of the model and in order to provide an interesting comparison, we evaluate a traditional string-based language model on the same data.

2 Related Work

The idea of using a language model based on deep syntax is not new to SMT. Shen et al. (2008) use a dependency-based language model in a string to dependency tree SMT system for Chinese-English translation, using information from the deeper structure about dependency relations between words, in addition to the position of the words in the string, including information about whether context words were positioned on the left or right of a word. Bojar and Hajič (2008) use a deep syntax language model in an English-Czech dependency tree-to-tree transfer system, and include three separate bigram language models: a reverse, direct and joint model. The model in our evaluation is similar to their direct bigram model, but is not restricted to bigrams.

Riezler and Maxwell (2006) use a trigram deep syntax language model in German-English dependency tree-to-tree transfer to re-rank decoder output. The language model of Riezler and Maxwell (2006) is similar to the model in our evaluation, but differs in that it is restricted to a trigram model trained on LFG f-structures. In addition, as language modeling is not the main focus of their work, they provide little detail on the language model they use, except to say that it is based on *"log-probability of strings of predicates from root to frontier of target f-structure, estimated from predicate trigrams in English f-structures"* (Riezler and Maxwell, 2006). An important property of LFG f-structures (and deep syntactic structures in general) was possibly overlooked here. F-structures can contain more than one path of predicates from the root to a frontier that include the same ngram, and this occurs when the underlying graph includes unary branching followed by branching with arity greater than one. In such cases, the language model probability as described in Riezler and Maxwell (2006) is incorrect as the probability of these ngrams will be included multiple times. In our definition of a deep syntax language model, we ensure that such duplicate ngrams are omitted in training and testing. In addition, Wu (1998) use a bigram deep syntax language model in a stochastic inversion transduction grammar for English to Chinese. None of the related research we discuss here has included an evaluation of the deep syntax language model they employ in isolation from the MT system, however.

3 Deep Syntax

The deep syntax language model we describe is not restricted to any individual theory of deep syntax. For clarity, however, we restrict our examples to LFG, which is also the deep syntax theory we use for our evaluation. The Lexical Functional Grammar (LFG) (Kaplan and Bresnan, 1982; Kaplan, 1995; Bresnan, 2001; Dalrymple, 2001) functional structure (f-structure) is an attribute-value encoding of bi-lexical labeled dependencies, such as *subject*, *object* and *adjunct* for example, with morpho-syntactic atomic attributes encoding information such as *mood* and *tense* of verbs, and *person*, *number* and *case* for nouns. Figure 1 shows the LFG f-structure for English sentence *"Today congress passed Obama's health care bill."*[1]

Encoded within the f-structure is a directed graph and our language model uses a simplified acyclic unlabeled version of this graph. Figure 1(b) shows the graph structure encoded within the f-structure of Figure 1(a). We discuss the simplification procedure later in Section 5.

[1] Morpho-syntactic information/ atomic features are omitted from the diagram.

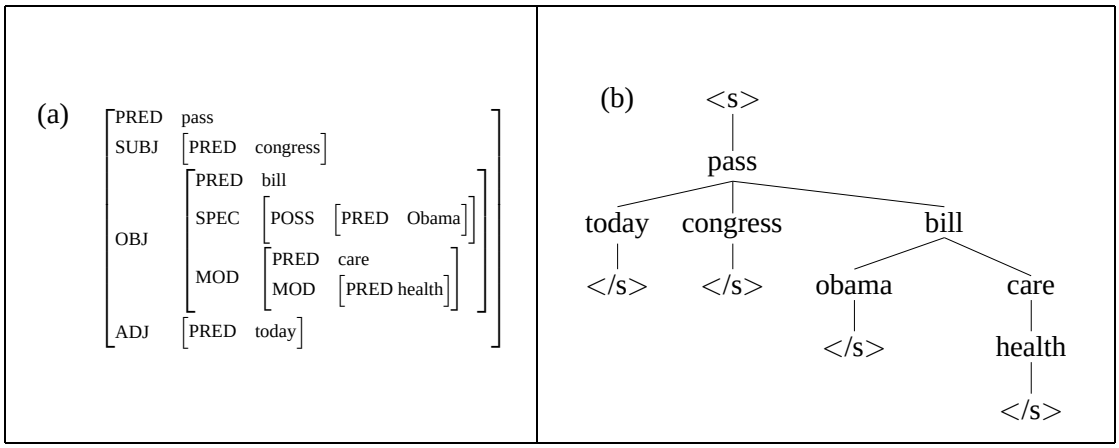


Figure 1: "*Today congress passed Obama's health care bill.*"

4 Language Model

We use a simplified approximation of the deep syntactic structure, d_e, that encodes the unlabeled dependencies between the words of the sentence, to estimate a deep syntax language model probability. Traditional string-based language models combine the probability of each word in the sentence, w_i, given its preceding context, the sequence of words from w_1 to w_{i-1}, as shown in Equation 1.

$$p(w_1, w_2, ..., w_l) = \prod_{i=1}^{l} p(w_i|w_1, ..., w_{i-1}) \quad (1)$$

In a similar way, a deep syntax language model probability combines the probability of each word in the structure, w_i, given its context within the structure, the sequence of words from w_r, the head of the sentence, to $w_{m(i)}$, as shown in Equation 2, with function m used to map the index of a word in the structure to the index of its head. [2]

$$p(d_e) = \prod_{i=1}^{l} p(w_i|w_r, ..., w_{m(m(i))} w_{m(i)}) \quad (2)$$

In order to combat data sparseness, we apply the Markov assumption, as is done in traditional string-based language modeling, and simplify the probability by only including a limited length of history when estimating the probability of each word in the structure. For example, a trigram deep syntax language model conditions the probability of each word on the sequence of words consisting of *the head of the head of the word* followed by *the head of the word* as follows:

$$p(d_e) = \prod_{i=1}^{l} P(w_i|w_{m(m(i))}, w_{m(i)}) \quad (3)$$

In addition, similar to string-based language modeling, we add a start symbol, <s>, at the root of the structure and end symbols, </s>, at the leaves to include the probability of a word being the head of the sentence and the probability of words occurring as leaf nodes in the structure. Figure 2(a) shows an example of how a trigram deep syntax language model probability is computed for the example sentence in Figure 1(a).

5 Simplified Approximation of the Deep Syntactic Representation

We describe the deep syntactic structure, d_e, as an approximation since a parser is employed to automatically produce it and there is therefore no certainty that we use the actual/correct deep syntactic representation for the sentence. In addition, the function m requires that each node in the structure has exactly one head, however, structure-sharing can occur within deep syntactic structures resulting in a single word legitimately having two heads. In such cases we use a simplification of the graph in the deep syntactic structure. Figure 3 shows an f-structure in which the *subject*

[2] We refer to the lexicalized nodes in the dependency structure as *words*, alternatively the term *predicate* can be used.

(a)	Deep Syntax LM	(b)	Traditional LM
$p(e)$	$\approx$ $p($ pass | <s>$)*$	$p(e)$	$\approx$ $p($ passed | today congress $)*$
	$p($ today | <s> pass $)*$		$p($ today | <s>$)*$
	$p($</s> | pass today $)*$		
	$p($ congress | <s> pass $)*$		$p($ congress | <s> today $)*$
	$p($</s> | pass congress $)*$		
	$p($ bill | <s> pass $)*$		$p($ bill | health care $)*$
	$p($ obama | pass bill $)*$		$p($ obama | congress passed $)*$
	$p($</s> | bill obama $)*$		
	$p($ care | pass bill $)*$		$p($ care | s health $)*$
	$p($ health | bill care $)*$		$p($ health | ' s $)*$
	$p($</s> | care health $)$		
			$p($ ' | passed Obama $)*$
			$p($ s | obama ' $)*$
			$p($. | care bill $)*$
			$p($</s> | bill . $)$

Figure 2: Example Comparison of Deep Syntax and Traditional Language Models

of both *like*, *be* and *president* is *hillary*. In our simplified structure, the dependency relations between *be* and *hillary* and *president* and *hillary* are dropped. We discuss how we do this later in Section 6. Similar to our simplification for structure sharing, we also simplify structures that contain cycles by discarding edges that cause loops in the structure.

6 Implementation

SRILM (Stolcke, 2002) can be used to compute a language model from ngram counts (the *-read* option of the *ngram-count* command). Implementation to train the language model, therefore, simply requires accurately extracting counts from the deep syntax parsed training corpus. To simplify the structures to acyclic graphs, nodes are labeled with an increasing index number via a depth first traversal. This allows each arc causing a loop in the graph or argument sharing to be identified by a simple comparison of index numbers, as the index number of its start node will be greater than that of its end node. The algorithm we use to extract ngrams from the dependency structures is straightforward: we simply carry out a depth-first traversal of the graph to construct paths of words that stretch from the root of the graph to words

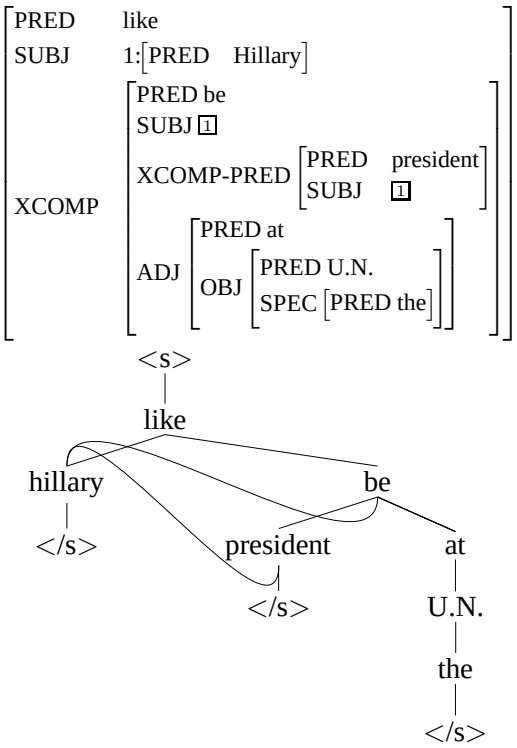


Figure 3: "*Hillary liked being president at the U.N.*"

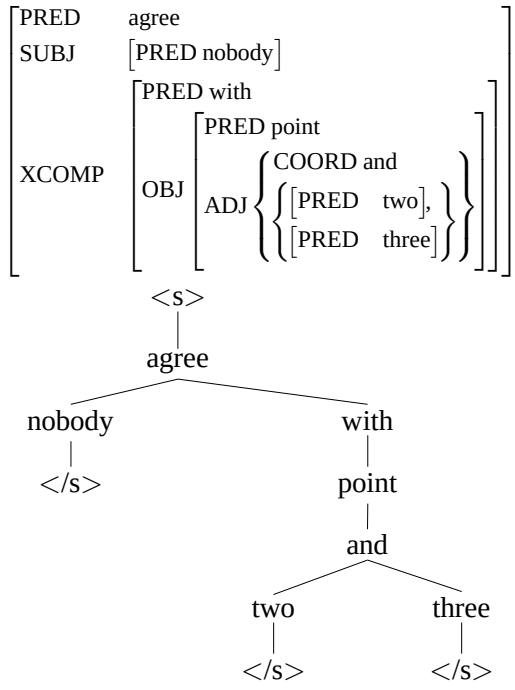


Figure 4: *"Nobody agreed with points two and three."*

at the leaves and then extract the required order ngrams from each path. As mentioned earlier, some ngrams can belong to more than one path. Figure 4 shows an example structure containing unary branching followed by binary branching in which the sequence of symbols and words *"<s> agree with point and"* belong to the path ending in *two </s>* and *three </s>*. In order to ensure that only distinct ngrams are extracted we assign each word in the structure a unique id number and include this in the extracted ngrams. Paths are split into ngrams and duplicate ngrams resulting from their occurrence in more than one path are discarded. Its also possible for ngrams to legitimately be repeated in a deep structure, and in such cases we do not discard these ngrams. Legitimately repeating ngrams are easily identified as the id numbers attached to words will be different.

7 Deep Syntax and Lexical Choice in SMT

Correct lexical choice in machine translation is extremely important and PB-SMT systems rely on the language model to ensure, that when two phrases are combined with each other, that the model can rank combined phrases that are fluent higher than less fluent combinations. Conditioning the probability of each word on its deep context has the potential to provide a more meaningful context than the local context within the string. A comparison of the probabilities of individual words in the deep syntax model and traditional language model in Figure 2 clearly shows this. For instance, let us consider how the language model in a German to English SMT system is used to help rank the following two translations *today congress passed ...* and *today convention passed ...* (the word *Kongress* in German can be translated into either *congress* or *convention* in English). In the deep syntax model, the important competing probabilities are (i) $p(congress|\text{<s>}pass)$ and (ii) $p(convention|\text{<s>}pass)$, where (i) can be interpreted as the probability of the word *congress* modifying *pass* when *pass* is the head of the entire sentence and, similarly (ii) the probability of the word *convention* modifying *pass* when *pass* is the head of the entire sentence. In the traditional string-based language model, the equivalent competing probabilities are (i) $p(congress|\text{<s>}today)$, the probability of *congress* following *today* when *today* is the start of the sentence and (ii) $p(convention|\text{<s>}today)$, probability of convention following *today* when *today* is the start of the sentence, showing that the deep syntax language model is able to use more meaningful context for good lexical choice when estimating the probability of words *congress* and *convention* compared to the traditional language model.

In addition, the deep syntax language model will encounter less data sparseness problems for some words than a string-based language model. In many languages words occur that can legitimately be moved to different positions within the string without any change to dependencies between words. For example, sentential adverbs in English, can legitimately change position in a sentence, without affecting the underlying dependencies between words. The word *today* in *"Today congress passed Obama's health bill"*

can appear as *"Congress passed Obama's health bill today"* and *"Congress today passed Obama's health bill"*. Any sentence in the training corpus in which the word *pass* is modified by *today* will result in a bigram being counted for the two words, regardless of the position of *today* within each sentence.

In addition, some surface form words such as auxiliary verbs for example, are not represented as predicates in the deep syntactic structure. For lexical choice, its not really the choice of auxiliary verbs that is most important, but rather the choice of an appropriate lexical item for the main verb (that belongs to the auxiliary verb). Omitting auxiliary verbs during language modeling could aid good lexical choice, by focusing on the choice of a main verb without the effect of what auxiliary verb is used with it.

For some words, however, the probability in the string-based language model provides as good if not better context than the deep syntax model, but only for the few words that happen to be preceded by words that are important to its lexical choice, and this reinforces the idea that SMT systems can benefit from using both a deep syntax and string-based language model. For example, the probability of *bill* in Figures 2(a) and 2(b) is computed in the deep syntax model as $p(bill|$ <s> $pass)$ and in the string-based model using $p(bill|health\ care)$, and for this word the local context seems to provide more important information than the deep context when it comes to lexical choice. The deep model nevertheless adds some useful information, as it includes the probability of *bill* being an argument of *pass* when *pass* is the head of a sentence.

In traditional language modeling, the special start symbol is added at the beginning of a sentence so that the probability of the first word appearing as the first word of a sentence can be included when estimating the probability. With similar motivation, we add a start symbol to the deep syntactic representation so that the probability of the head of the sentence occurring as the head of a sentence can be included. For example, $p(be|$ <s>) will have a high probability as the verb *be* is the head of many sentences of English, whereas $p(colorless|$ <s>) will have a low probability since it is unlikely to occur as the head. We also add end symbols at the leaf nodes in the structure to include the probability of these words appearing at that position in a structure. For instance, a noun followed by its determiner such as $p(</s>|attorney\ a)$ would have a high probability compared to a conjunction followed by a verb $p(</s>|and\ be)$.

8 Evaluation

We carry out an experimental evaluation to investigate the potential of the deep syntax language model we describe in this paper independently of any machine translation system. We train a 5-gram deep syntax language model on 7M English f-structures, and evaluate it by computing the perplexity and ngram coverage statistics on a held-out test set of parsed fluent English sentences. In order to provide an interesting comparison, we also train a traditional string-based 5-gram language model on the same training data and test it on the same held-out test set of English sentences. A deep syntax language model comes with the obvious disadvantage that any data it is trained on must be in-coverage of the parser, whereas a string-based language model can be trained on any available data of the appropriate language. Since parser coverage is not the focus of our work, we eliminate its effects from the evaluation by selecting the training and test data for both the string-based and deep syntax language models on the basis that they are in fact in-coverage of the parser.

8.1 Language Model Training

Our training data consists of English sentences from the WMT09 monolingual training corpus with sentence length range of 5-20 words that are in coverage of the parsing resources (Kaplan et al., 2004; Riezler et al., 2002) resulting in approximately 7M sentences. Preparation of training and test data for the traditional language model consisted of tokenization and lower casing. Parsing was carried out with XLE (Kaplan et al., 2002) and an English LFG grammar (Kaplan et al., 2004; Riezler et al., 2002). The parser produces a packed representation of all possible parses according to the LFG grammar and we select only the single best parse for language model training by means of a disambiguation model (Kaplan et

Corpus	Tokens	Ave. Tokens per Sent.	Vocab
strings	138.6M	19	345K
LFG lemmas/predicates	118.4M	16	280K

Table 1: Language model statistics for string-based and deep syntax language models, statistics are for string tokens and LFG lemmas for the same set of 7.29M English sentences

al., 2004; Riezler et al., 2002). Ngrams were automatically extracted from the f-structures and lowercased. SRILM (Stolcke, 2002) was used to compute both language models. Table 1 shows statistics on the number of words and lemmas used to train each model.

8.2 Testing

The test set consisted of 789 sentences selected from WMT09 additional development sets[3] containing English Europarl text and again was selected on the basis of sentences being in-coverage of the parsing resources. SRILM (Stolcke, 2002) was used to compute test set perplexity and ngram coverage statistics for each order model.

Since the deep syntax language model adds end of sentence markers to leaf nodes in the structures, the number of (so-called) end of sentence markers in the test set for the deep syntax model is much higher than in the string-based model. We therefore also compute statistics for each model when end of sentence markers are omitted from training and testing. [4] In addition, since the vast majority of punctuation is not represented as predicates in LFG f-structures, we also test the string-based language model when punctuation has been removed.

8.3 Results

Table 2 shows perplexity scores and ngram coverage statistics for each order and type of language model. Note that perplexity scores for the string-based and deep syntax language models are not directly comparable because each model has a different vocabulary. Although both models train on an identical set of sentences, the data is in a different format for each model, as the string-based model is trained on surface form tokens, whereas the deep syntax model uses lemmas. Ngram coverage statistics provide a better comparison.

Unigram coverage for all models is high with all models achieving close to 100% coverage on the held-out test set. Bigram coverage is highest for the deep syntax language model when eos markers are included (94.71%) with next highest coverage achieved by the string-based model that includes eos markers (93.09%). When eos markers are omitted bigram coverage goes down slightly to 92.44% for the deep syntax model and to 92.83% for the string-based model, and when punctuation is also omitted from the string-based model, coverage goes down again to 91.57%.

Trigram coverage statistics for the test set maintain the same rank between models as in the bigram coverage, from highest to lowest as follows: DS+eos at 64.71%, SB+eos at 58.75%, SB-eos at 56.89%, DS-eos at 53.67%, SB-eos-punc at 53.45%. For 4-gram and 5-gram coverage a similar coverage ranking is seen, but with DS-eos (4gram at 17.17%, 5gram at 3.59%) and SB-eos-punc (4gram at 20.24%, 5gram at 5.76%) swapping rank position.

8.4 Discussion

Ngram coverage statistics for the DS-eos and SB-eos-punc models provide the fairest comparison, with the deep syntax model achieving higher coverage than the string-based model for bigrams (+0.87%) and trigrams (+0.22%), marginally lower coverage coverage of unigrams (-0.02%) and lower coverage of 4-grams (-3.07%) and 5-grams (2.17%) compared to the string-based model.

Perplexity scores for the deep syntax model when eos symbols are included are low (79 for the 5gram model) and this is caused by eos markers

[3] test2006.en and test2007.en

[4] When we include end of sentence marker probabilities we also include them for normalization, and omit them from normalization when their probabilities are omitted.

	1-gram		2-gram		3-gram		4-gram		5-gram	
	cov.	ppl	cov.	ppl	cov.	ppl	cov.	ppl	cov.	ppl
SB-eos	99.61%	1045	92.83%	297	56.89%	251	23.32%	268	7.19%	279
SB-eos-punc	99.58%	1357	91.57%	382	53.45%	327	20.24%	348	5.76%	360
DS-eos	99.56%	1005	92.44%	422	53.67%	412	17.17%	446	3.59%	453
SB+eos	99.63%	900	93.09%	227	58.75%	194	25.48%	207	8.35%	215
DS+eos	99.70%	211	94.71%	77	64.71%	73	29.86%	78	8.75%	79

Table 2: Ngram coverage and perplexity (ppl) on held-out test set. Note: DS = deep syntax, SB string-based, eos = end of sentence markers

in the test set in general being assigned relatively high probabilities by the model, and since several occur per sentence, the perplexity increases when the are omitted (453 for the 5gram model).

Tables 3 and 4 show the most frequently encountered trigrams in the test data for each type of model. A comparison shows how different the two models are and highlights the potential of the deep syntax language model to aid lexical choice in SMT systems. Many of the most frequently occurring trigram probabilities for the deep syntax model are for arguments of the main verb of the sentence, conditioned on the main verb, and including such probabilities in a system could improve fluency by using information about which words are in a dependency relation together explicitely in the model. In addition, a frequent trigram in the held-out data is *<s> be also*, where the word *also* is a sentential adverb modifying *be*. Trigrams for sentential adverbs are likely to be less effected by data sparseness in the deep syntax model compared to the string-based model which could result in the deep syntax model improving fluency with respect to combinations of main verbs and their modifying adverbs. The most frequent trigram in the deep syntax test set is *<s> and be*, in which the head of the sentence is the conjunction *and* with argument *be*. In this type of syntactic construction in English, its often the case that the conjunction and verb will be distant from each other in the sentence, for example: *Nobody was there except the old lady and without thinking we quickly left.* (where *was* and *and* are in a dependency relation). Using a deep syntax language model could therefore improve lexical choice for such words, since they are too distant for a string-based model.

3-gram	No. Occ.	Prob.
<s> and be	42	0.1251
<s> be this	21	0.0110
<s> must we	19	0.0347
<s> would i	19	0.0414
<s> be in	17	0.0326
<s> be that	14	0.0122
be debate the	13	0.0947
<s> be debate	13	0.0003
<s> can not	12	0.0348
<s> and president	11	0.0002
<s> would like	11	0.0136
<s> would be	11	0.0835
<s> be also	10	0.0075

Table 3: Most frequent trigrams in test set for deep syntax model

9 Conclusions

We presented a comparison of a deep syntax language and traditional string-based language model. Results showed that the deep syntax language model achieves similar ngram coverage to the string-based model on a held out test set. We highlighted the potential of integrating such a model into SMT systems for improving lexical choice by using a deeper context for probabilities of words compared to a string-based model.

References

Bojar, Ondřej, Jan Hajič. 2008. Phrase-Based and Deep Syntactic English-to-Czech Statistical Machine Translation. In *Proceedings of the third Work-*

3-gram	No. Occ.	Prob.
mr president ,	40	0.5385
<s> this is	25	0.1877
by the european	20	0.0014
the european union	18	0.1096
<s> it is	16	0.1815
the european parliament	15	0.0252
would like to	15	0.4944
<s> i would	15	0.0250
<s> that is	14	0.1094
i would like	14	0.0335
and gentlemen ,	13	0.1005
ladies and gentlemen	13	0.2834
<s> we must	12	0.0120
should like to	12	0.1304
i should like	11	0.0089
, ladies and	11	0.5944
, it is	10	0.1090

Table 4: Most frequent trigrams in test set for string-based model

shop on Statistical Machine Translation, Columbus, Ohio.

Bresnan, Joan. 2001. *Lexical-Functional Syntax.*, Blackwell Oxford.

Chiang, David. 2007. Hierarchical Phrase-based Models of Translation In *Computational Linguistics*, No. 33:2.

Chiang, David. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 263-270, Ann Arbor, Michigan.

Dalrymple, Mary. 2001. *Lexical Functional Grammar*, Academic Press, San Diego, CA; London.

Kaplan, Ronald, Stefan Riezler, Tracy H. King, John T. Maxwell, Alexander Vasserman. 2004. Speed and Accuracy in Shallow and Deep Stochastic Parsing. In *Proceedings of Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics Meeting*, Boston, MA.

Kaplan, Ronald M., Tracy H. King, John T. Maxwell. 2002. Adapting Existing Grammars: the XLE Experience. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING) 2002*, Taipei, Taiwan.

Kaplan, Ronald M. 1995. The Formal Architecture of Lexical Functional Grammar. In *Formal Issues in Lexical Functional Grammar*, ed. Mary Dalrymple, pages 7-28, CSLI Publications, Stanford, CA.

Kaplan, Ronald M., Joan Bresnan. 1982. Lexical Functional Grammar, a Formal System for Grammatical Represenation. In J. Bresnan, editor, *The Mental Representation of Grammatical Relations*, 173-281, MIT Press, Cambridge, MA.

Koehn, Philipp, Hieu Hoang. 2007. Factored Translation Models. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 868-876.

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

Koehn, Philipp 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. In *Proceedings of the tenth Machine Translation Summit*.

Koehn, Philipp, Franz Josef Och, Daniel Marcu. 2003. Statistical Phrase-based Translation. In *Proceedings of Human Language Technology and North American Chapter of the Association for Computational Linguistics Conference*, 48-54.

Riezler, Stefan, John T. Maxwell III. 2006. Grammatical Machine Translation. In *Proceedings of HLT-ACL*, pages 248-255, New York.

Riezler, Stefan, Tracy H. King, Ronald M. Kaplan, Richard Crouch, John T. Maxwell, Mark Johnson. 2002. Parsing the Wall Street Journal using Lexical Functional Grammar and Discriminitive Estimation Techniques . (grammar version 2005) In *Proceedings of the 40th ACL*, Philadelphia.

Shen, Libin, Jinxi Xu, Ralph Weischedel. 2008. A New String-to-Dependency Machine Translation Algorithm with a Target Dependency Language Model. *Proceedings of ACL-08: HLT*, pages 577-585.

Stolcke, Andreas. 2002. SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, Denver, Colorado.

Dekai, Wu, Hongsing Wong. 1998. Machine Translation with a Stochastic Grammatical Channel. In *Proceedings of the 36th ACL and 17th COLING*, Montreal, Quebec.

Author Index