An Empirical Study of Translation Rule Extraction with Multiple Parsers

Tong Xiao[†‡], Jingbo Zhu[†‡], Hao Zhang[†], Muhua Zhu[†‡]

[†]Natural Language Processing Lab., Northeastern University
[‡]Key Laboratory of Medical Image Computing, Ministry of Education
{xiaotong,zhujingbo}@mail.neu.edu.cn
zhanghao@ics.neu.edu.cn, zhumuhua@gmail.com

Abstract

Translation rule extraction is an important issue in syntax-based Statistical Machine Translation (SMT). Recent studies show that rule coverage is one of the key factors affecting the success of syntax-based systems. In this paper, we first present a simple and effective method to improve rule coverage by using multiple parsers in translation rule extraction, and then empirically investigate the effectiveness of our method on Chinese-English translation tasks. Experimental results show that extracting translation rules using multiple parsers improves a string-to-tree system by over 0.9 BLEU points on both NIST 2004 and 2005 test corpora.

1 Introduction

Recently various syntax-based models have been extensively investigated in Statistical Machine Translation (SMT), including models between source trees and target strings (Quirk et al., 2005; Liu et al., 2006; Huang et al., 2006), source strings and target trees (Yamada and Knight, 2001; Galley et al., 2006; Shen et al., 2008), or source trees and target trees (Eisner, 2003; Ding and Palmer, 2005; Cowan et al., 2006; Zhang et al., 2008; Liu et al., 2009). In these models, automatic extraction of translation rules is an important issue, in which translation rules are typically extracted using parse trees on source/target-language side or both sides of the bilingual text. Exploiting the syntactic information encoded in translation rules, syntax-based systems have shown to achieve comparable performance with phrase-based systems, even outperform them in some cases (Marcu et al., 2006).

Among all the factors contributing to the success of syntax-based systems, *rule coverage* has been proved to be an important one that affects the translation accuracy of syntax-based systems (DeNeefe et al., 2007; Shen et al., 2008). However, these systems suffer from a problem that translation rules are extracted using only 1-best parse tree generated by a single parser, which generally results in relatively low rule coverage due to the limited scope in rule extraction (Mi and Huang, 2008). To alleviate this problem, a straightforward solution is to enlarge the scope of rule extraction, and obtain translation rules by using a group of diversified parse trees instead of a single parse tree. For example, Mi and Huang (2008) used *k*-best parses and forest to extract translation rules for improving the rule coverage in their forest-based SMT system, and achieved promising results. However, most previous work used the parse trees generated by only one parser, which still suffered somewhat from the relatively low diversity in the outputs of a single parser.

Addressing this issue, we investigate how to extract diversified translation rules using multiple parsers. As different parsers (or parsing models) can provide us with parse trees having relatively large diversity, we believe that it is beneficial to employ multiple different parsers to obtain diversified translation rules and thus enlarge the rule coverage. Motivated by this idea, we propose a simple and effective method to improve rule coverage by using multiple parsers

Coling 2008: Poster Volume, pages 1345–1353,
Beijing, August 2010

in rule extraction. Furthermore, we conduct an empirical study to investigate the effectiveness of our method on Chinese-English translation in a string-to-tree system. Experimental results show that our method improves the baseline system by over 0.9 BLEU points on both NIST 2004 and 2005 test corpora, even achieves a +1 BLEU improvement when working with the *k*-best extraction method. More interestingly, we observe that the MT performance is not very sensitive to the parsing performance of the parsers used in rule extraction. Actually, the MT system does not show different preferences for different parsers.

2 Related Work

In machine translation, some efforts have been made to improve rule coverage and advance the performance of syntax-based systems. For example, Galley et al. (2006) proposed the idea of rule composing which composes two or more rules with shared states to form a larger, composed rule. Their experimental results showed that the rule composing method could significantly improve the translation accuracy of their syntax-based system. Following Galley et al. (2006)'s work, Marcu et al. (2006) proposed SPMT models to improve the coverage of phrasal rules, and demonstrated that the system performance could be further improved by using their proposed models. Wang et al. (2007) described a binarization method that binarized parse trees to improve the rule coverage on non-syntactic mappings. DeNeefe et al. (2007) analyized the phrasal coverage problem, and compared the phrasal coverage as well as translation accuracy for various rule extraction methods (Galley et al., 2006; Marcu et al., 2006; Wang et al., 2007).

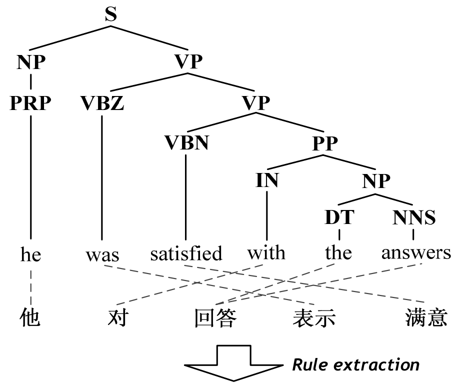
As another research direction, some work is focused on enlarging the scope of rule extraction to improve rule coverage. For example, (Venugopal et al., 2008) and (Mi and Huang, 2008) extracted rules from the *k*-best parses and forest generated by a single parser to alleviate the problem of the limited scope of 1-best parse, and achieved promising results.

Our work differs from previous work in that we are concerned with obtaining diversified translation rules using multiple different parsers (or parsing models) instead of a single parser (or parsing model). It can be regarded as an enhancement of previous studies. As shown in the following parts of this paper, it works very well with the existing techniques, such as rule composing (Galley et al., 2006), SPMT models (Marcu et al., 2006) and rule extraction with *k*-best parses (Venugopal et al., 2008).

3 Translation Rule Extraction

In this work, the issue of translation rule extraction is studied in the string-to-tree model proposed by Galley et al. (2006). We choose this model because it has been shown to be one of the state-of-the-art syntax-based models, and has been adopted in the most successful systems in NIST 2009 MT evaluation.

Typically, (string-to-tree) translation rules are learned from the word-aligned bilingual text whose target-side has been parsed using a syntactic parser. As the basic unit of translation, a translation rule consists of sequence words or variables in the source language, and a syntax tree in the target language having words (terminals) and variables (non-terminals) at leaves. Figure 1 shows the translation rules extracted from a word-aligned sentence pair with a target-side parse tree.



r_1: 他 → **PRP** (he)

r_2: 对 → **IN** (with)

r_3: 回答 → **NP** (**DT**(the) **NNS**(answers))

r_4: 表示 → **VBZ** (was)

r_5: 满意 → **VBN** (satisfied)

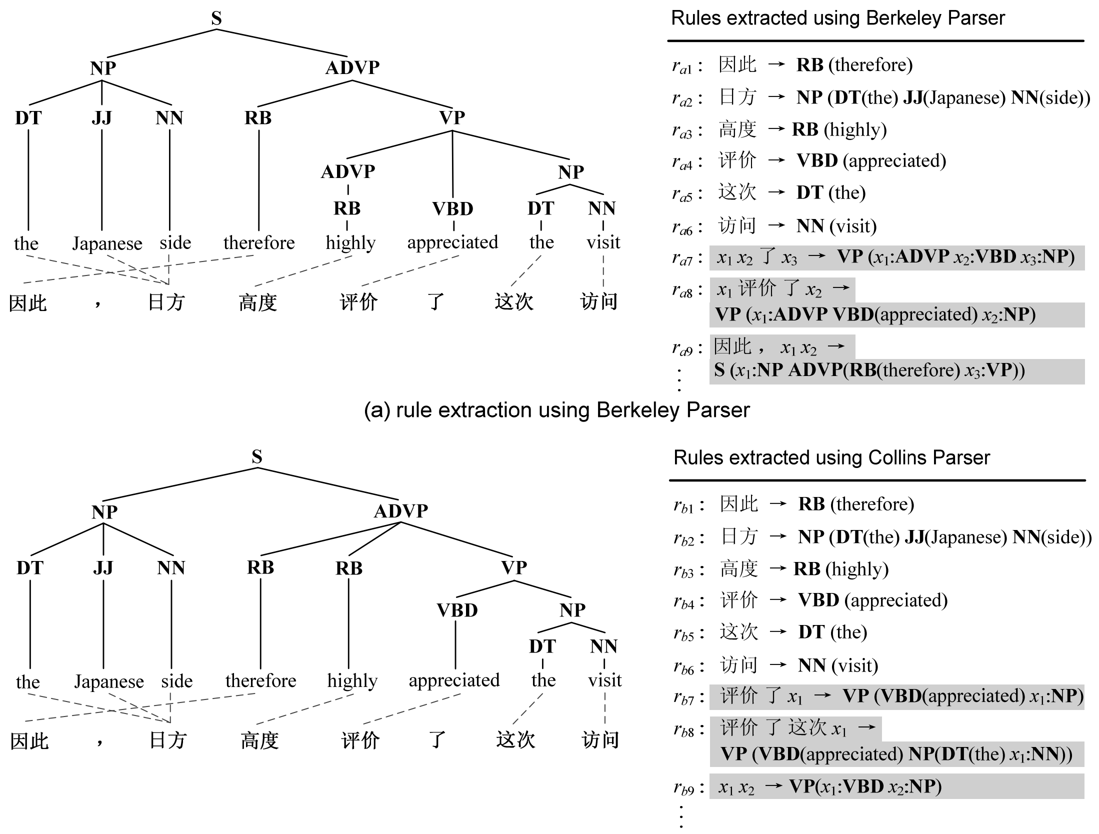
r_6: $x_1\, x_2$ → **PP** (x_1:**IN** x_2:**NP**)

r_7: 对 x_1 → **PP** (**IN**(with) x_1:**NP**)

r_8: $x_1\, x_2$ 表示 满意 →
S (x_1:**NP** **VP**(**VBZ**(was) **VP**(**VBN**(satisfied) x_2:**PP**)))

⋮

Figure 1: Translation rules extracted from a string-tree pair.


S
NP
ADVP
DT
JJ
NN
RB
VP
ADVP
RB
VBD
NP
DT
NN
the
Japanese
side
therefore
highly
appreciated
the
visit
因此
，
日方
高度
评价
了
这次
访问
Rules extracted using Berkeley Parser
r_{a1}: 因此 → RB (therefore)
r_{a2}: 日方 → NP (DT(the) JJ(Japanese) NN(side))
r_{a3}: 高度 → RB (highly)
r_{a4}: 评价 → VBD (appreciated)
r_{a5}: 这次 → DT (the)
r_{a6}: 访问 → NN (visit)
r_{a7}: x_1 x_2 了 x_3 → VP (x_1:ADVP x_2:VBD x_3:NP)
r_{a8}: x_1 评价 了 x_2 → VP (x_1:ADVP VBD(appreciated) x_2:NP)
r_{a9}: 因此 ， x_1 x_2 → S (x_1:NP ADVP(RB(therefore) x_3:VP))
⋮

(a) rule extraction using Berkeley Parser

S
NP
ADVP
DT
JJ
NN
RB
RB
VP
VBD
NP
DT
NN
the
Japanese
side
therefore
highly
appreciated
the
visit
因此
，
日方
高度
评价
了
这次
访问
Rules extracted using Collins Parser
r_{b1}: 因此 → RB (therefore)
r_{b2}: 日方 → NP (DT(the) JJ(Japanese) NN(side))
r_{b3}: 高度 → RB (highly)
r_{b4}: 评价 → VBD (appreciated)
r_{b5}: 这次 → DT (the)
r_{b6}: 访问 → NN (visit)
r_{b7}: 评价 了 x_1 → VP (VBD(appreciated) x_1:NP)
r_{b8}: 评价 了 这次 x_1 → VP (VBD(appreciated) NP(DT(the) x_1:NN))
r_{b9}: x_1 x_2 → VP(x_1:VBD x_2:NP)
⋮

(b) rule extraction using Collins Parser (Model 2)

Figure 2: Rule extraction using two different parsers (Berkeley Parser and Collins Parser). The shaded rectangles denote the translation rules that can be extracted from the parse tree generated by one parser but cannot be extracted from the parse tree generated by the other parser.

To obtain basic translation rules, the (minimal) GHKM extraction method proposed in (Galley et al, 2004) is utilized. The basic idea of GHKM extraction is to compute the set of the minimally-sized translation rules that can explain the mappings between source-language string and target-language tree while respecting the alignment and reordering between the two languages. For example, from the string-tree pair shown at the top of Figure 1, we extract the minimal GHKM translation rules $r_{1\text{-}6}$. In addition to GHKM extraction, the SPMT models (Marcu et al., 2006) are employed to obtain *phrasal rules* that are not covered by GHKM extraction. For example, rule r_8 in Figure 1 is a SPMT rule that is not obtained in GHKM extraction. Finally, the rule composing method (Galley et al., 2006) is used to compose two or more minimal GHKM or SPMT rules having shared states to form larger rules. For example, rule r_7 in Figure 1 is generated by composing rules r_2 and r_6.

4 Differences in Coverage between Rule Extractions with Different Parsers

As described above, translation rule extraction relies on the outputs (parse trees) of parsers. As different parsers generally have large diversity between their outputs, rule extractions with different parsers generally result in very different sets of rules. For example, Figure 2 shows the rule extractions on a word-aligned sentence pair having two target-trees generated by Berkeley Parser and Collins Parser, respectively. It is observed that Figure 2 (a) and (b) cover different sets of rule due to the different target-trees used in rule extraction. Particularly, well-formed rules $r_{a7\text{-}a9}$ are extracted in Figure 2 (a), while they do not appear in Figure 2 (b). Also, rules $r_{b7\text{-}b9}$ in Figure 2 (b) have the similar situation. This observation gives us an intuition that there is a "complementarity" between the rules extracted using different parsers.

We also conduct a quantitative study to investigate the impact of using different parsers (Berkeley Parser and Collins Parser) on rule coverage. Tables 1 shows the statistics of the rules extracted from 370K Chinese-English parallel sentence pairs[1] using the method described in Section 3. In addition to the total number of rules extracted, the numbers of *phrasal rules* and *useful rules* are also reported to indicate the rule coverage of a rule set. Here *phrasal rule* refers to the rule whose source-side and the yield of its target-side contains only one phrase each, with optional surrounding variables. According to (DeNeefe et al., 2007), the number of phrasal rules is a good indicator of the coverage of a rule set. *useful rule* refers to the rule that can be applied when decoding the test sentences[2]. As shown in Table 1, the two resulting rule sets only have about 70% overlaps (Column 4), and the rule coverage increases by about 20% when we combine them together (Column 5). This finding confirms that the rule coverage can be improved by using multiple different parsers in rule extraction.

	# of rules	# of phrasal rules	# of useful rules
Berkeley	3,538,332	2,515,243	549,783
Collins	3,526,166	2,481,195	553,893
Overlap	2,542,380	1,907,521	386,983
Union	4,522,118	3,088,920	716,693

Table 1: Comparison of rule coverage between different rule sets.

5 Translation Rule Extraction with Multiple Parsers

5.1 Rule Extraction Algorithm

Motivated by the above observations, we propose a rule extraction method to improve the rule coverage by using multiple parsers.

Let <*f*, *e*, *a*> be a tuple of <source sentence, target sentence, bi-directional word alignments>, and $\{P_1, ..., P_N\}$ be N syntactic parsers in target-language. The following pseudocode formulizes the algorithm for extracting translation rules from <*f*, *e*, *a*> using parsers $\{P_1, ..., P_N\}$, where $P_i(e)$ returns the parse tree generated by the *i*-th parser P_i. Function GENERATERULES() computes the set of rules for <*f*, t_i, *a*> by using various rule extraction methods, such as the method described in Section 3.

Multi-Parser based Rule Extraction

```
Input: <f, e, a> and $P = \{P_1, ..., P_N\}$
Output: rule set $R$
Function MULTIPAREREXTRACTOIN(<f, e, a>, $P$ )
    for $i = 1$ to $N$ do                        ◁ for each parser
        $t_i = P_i(e)$                           ◁ target-tree
        $R_i$ = GENERATERULES ($f, t_i, a$)      ◁ rule extraction
        $R$.append($R_i$)
    return $R$
Function GENERATERULES ($f, t_i, a$ )
    return rules extracted from <$f, t_i, a$>
```

5.2 Learning Rule Probabilities

In multi-parser based rule extraction, more than one parse trees are used, and each of them is associated with a parsing confidence (e.g. generative probability of the tree). Ideally, if the parse trees used in rule extraction can be accurately weighted, the rule probabilities will be better estimated according to the parse weights, for example, the rules extracted from a parse tree having a low weight should be penalized accordingly in the estimation of rule probabilities. Unfortunately, the tree probabilities are generally incomparable between different parsers due to the different parsing models used and ways of implementation. Thus we cannot use the posterior probability of a rule's target-side to estimate the *fractional count* (Mi and Huang, 2008; Liu et al., 2009), which is used in maximum-likelihood estimation of rule probabilities. In this work, to simplify the problem, we assume that all the parsers have the same and maximum degrees of confidence on their outputs. For a rule r extracted from a string-tree pair, the count of r is defined to be:

$$c(r) = \frac{\sum_{i=1}^{N} \tau(r,i)}{N} \quad (1)$$

where $\tau(r,i)$ is 1 if r is extracted by using the *i*-th parser, otherwise 0.

[1] LDC2005T10, LDC2003E07, LDC2003E14 and LDC2005T06

[2] In this experiment, the test sentences come from NIST 2004 and 2005 MT evaluation sets. It should be noted that due to the pruning in decoding we cannot count the exact number of rules that can be used during decoding. In this work, we use an alternative – the number of rules matched with test sentences – to estimate an upper-bound approximately.

Following Mi and Huang (2008)'s work, three conditional rule probabilities are employed for experimenting with our method.

$$\Pr(r \mid root(r)) = \frac{c(r)}{\sum_{r':root(r')=root(r)} c(r)} \quad (2)$$

$$\Pr(r \mid lhs(r)) = \frac{c(r)}{\sum_{r':lhs(r')=lhs(r)} c(r)} \quad (3)$$

$$\Pr(r \mid rhs(r)) = \frac{c(r)}{\sum_{r':rhs(r')=rhs(r)} c(r)} \quad (4)$$

where *lhs*(*r*) and *rhs*(*r*) are the source-hand and target-hand sides of *r* respectively, and *root*(*r*) is the root of *r*'s target-tree.

5.3 Parser Indicator Features

For each rule, we define *N* indicator features (i.e. $\tau(r,i)$) to indicate a rule is extracted by using which parsers, and add them into the translation model. By training the feature weights with Minimum Error Rate Training (MERT), the system can learn preferences for different parsers automatically.

6 Experiments

The experiments are conducted on Chinese-English translation in a state-of-the-art string-to-tree SMT system.

6.1 Experimental Setup

Our bilingual data consists of 370K sentence pairs (9M Chinese words + 10M English words) which have been used in the experiment in Section 4. GIZA++ is employed to perform the bidirectional word alignment between the source and target sentences, and the final word alignment is generated using the inter-sect-diag-grow method. A 5-gram language model is trained on the target-side of the bilingual data and the Xinhua portion of English Gigaword corpus. The development data set comes from NIST MT 2003 evaluation set. To speed up MERT, sentences with more than 20 Chinese words are removed. The test sets are the NIST MT evaluation sets of 2004 and 2005.

Our baseline MT system is built based on the string-to-tree model proposed in (Galley et al., 2006). In this system, both of minimal GHKM (Galley et al., 2004) and SPMT rules (Marcu et al., 2006) are extracted from the bilingual corpus, and the composed rules are generated by composing two or three minimal GHKM and SPMT rules[3]. We use a CKY-style decoder with cube pruning (Huang and Chiang, 2007) and beam search to decode new Chinese sentences. By default, the beam size is set to 30. For integrating *n*-gram language model into decoding efficiently, rules containing more than two variables or source word sequences are binarized using the synchronous binarization method (Zhang et al., 2006; Xiao et al., 2009).

The system is evaluated in terms of the case-insensitive NIST version BLEU (using the shortest reference length), and statistical significant test is conducted using the re-sampling method proposed by Koehn (2004).

6.2 The Parsers

Four syntactic parsers are chosen for the experiments. They are Stanford Parser[4], Berkeley Parser[5], Collins Parser (Dan Bikel's reimplementation of Collins Model 2)[6] and Charniak Parser[7]. The former two are state-of-the-art non-lexicalized parsers, while the latter two are state-of-the-art lexicalized parsers. All the parsers are trained on sections 02-21 of the Wall Street Journal (WSJ) Treebank, and tuned on section 22. Table 2 summarizes the performance of the parsers.

Parser	Recall	Precision	F1
Stanford	86.29%	87.21%	86.75%
Berkeley	90.18%	90.45%	90.32%
Collins	89.14%	88.85%	88.99%
Charniak	89.99%	90.28%	90.13%

Table 2: Performance of the four parsers on section 23 of the WSJ Treebank.

We parse the target-side of the bilingual data using the four parsers individually. From the 1-best parses generated by these parsers, we obtain four baseline rule sets using the method described in Section 3, as well as the rule sets usi-

[3] Generally a higher baseline can be obtained by combining more (unit) rules. However, we find that using more composed rules does not affect the impact of using multiple parsers. Thus, we choose this setting in order to finish all experiments in time.

[4] http://nlp.stanford.edu/software/lex-parser.shtml

[5] http://code.google.com/p/berkeleyparser/

[6] http://www.cis.upenn.edu/~dbikel/download.html

[7] http://www.cs.brown.edu/people/ec/#software

	Rule set	Rule Coverage			BLEU4 (%)		
		# of rules	# of phrasal rules	# of useful rules	Dev.	MT04	MT05
Baseline	Stanford (S)	3,679 K	2,581 K	573 K	39.36	36.02	36.98
	Berkeley (B)	3,538 K	2,515 K	549 K	39.32	36.05	36.98
	Collins (Co)	3,526 K	2,481 K	553 K	39.16	36.07	36.91
	Charniak (Ch)	3,450 K	2,435 K	540 K	39.24	35.90	36.89
2 parsers	S + B	4,567 K	3,105 K	726 K	39.87+	36.57+	37.47+
	S + Co	4,734 K	3,202 K	752 K	39.94+	36.57+	37.52+
	S + Ch	4,764 K	3,258 K	751 K	40.01+	36.51	37.59+
	B + Co	4,522 K	3,088 K	716 K	39.84+	36.60+	37.46+
	B + Ch	4,562 K	3,129 K	717 K	39.81+	36.49	37.41
	Co + Ch	4,592 K	3,125 K	727 K	39.75	36.55+	37.43+
3 parsers	S + B + Co	5,331 K	3,543 K	852 K	40.14++	36.83++	37.78++
	S + B + Ch	5,380 K	3,590 K	854 K	40.05+	36.82++	37.70+
	S + Co + Ch	5,551 K	3,663 K	877 K	40.35++	36.70+	37.70+
	B + Co + Ch	5,294 K	3,544 K	840 K	40.04+	36.76+	37.65+
4	S + B + Co + Ch	6,005 K	3,940 K	958 K	40.28++	36.99++	37.89++

Table 5: Evaluation results. + or ++ = significantly better than all the baseline systems (using single parser) at the 95% or 99% confidence level.

	Stanford	Berkeley	Collins	Charniak
Stanford	100%	76.72%	73.32%	74.89%
Berkeley	76.72%	100%	75.69%	76.76%
Collins	73.32%	75.69%	100%	74.84%
Charniak	74.89%	76.76%	74.84%	100%

Table 3: Agreement between different parsers.

ng the multi-parser based rule extraction method. Before conducting primary experiments, we first investigate the differences between the 1-best outputs of the parsers. Table 3 shows the agreement between each pair of parsers. Here the degree of agreement shown in each cell is computed by using one parser's output as a good standard to evaluate the other parser's output in terms of F1 score, and a higher agreement score (i.e. F1 score) means that the 1-best outputs of the two parsers are more similar to each other. We see that the agreement scores between different parsers are always below 80%. This result reflects a large diversity in parse trees generated by different parsers, and thus confirms our observations in Section 4.

We also examine the "complementarity" between the baseline rule sets generated by using different parsers individually. Table 4 shows the results, where the degree of "complementarity" between two rule sets is defined as the percentage of the rules in one rule set that are not covered by the other rule set. It can be regarded as a measure of the disagreement between two rule sets, and a higher number indicates large "complementarity". For example, in Row 2, Column 3 (Table 4), "25.09%" means that 25.09% rules in the first rule set (using Stanford Parser) are not covered by the second rule set (using Berkeley Parser). Table 4 shows that there is always a disagreement of over 25% between different rule sets. These results indicate that using different parsers can lead to a relatively large "complementarity" between the rule sets.

	Stanford	Berkeley	Collins	Charniak
Stanford	0%	25.09%	29.91%	31.43%
Berkeley	27.98%	0%	27.90%	29.68%
Collins	32.84%	28.15%	0%	30.89%
Charniak	35.70%	31.43%	32.37%	0%

Table 4: Disagreement between the rule sets obtained using different parsers individually.

6.3 Evaluation of Translations

We then study the impact of multi-parser based rule extraction on translation accuracy. Table 5 shows the BLEU scores as well as the rule coverage for various rule extraction methods. We see, first of all, that the rule coverage is improved significantly by multi-parser based rule extraction. Compared to the baseline method (i.e. single-parser based rule extraction), the multi-parser based rule extraction achieves over 20% coverage improvements when only two parsers are used, even yields gains of over 50 percentage points when all the four parsers are used together. Also, BLEU score is improved by multi-parser based rule extraction. When two parsers are employed in rule extraction, there is generally a gain of over 0.4 BLEU points on both MT04 and MT05 test sets. Further improvements are achieved when more parsers are involved. On both test sets, using three parsers in rule extraction generally yields a +0.7 BLEU improvement, and using all the parsers together yields a +0.9 BLEU improvement which is the biggest improvement achieved in this set of experiment. All these results show that multi-parser based rule extraction is an effective way to improve the rule coverage as well as the BLEU score of the syntax-based MT system.

An interesting finding is that there seems no significant differences in BLEU scores between the baseline systems (using single parsers), though the parsing performance of the corresponding parsers is very different from each other. For example, the MT performance corresponding to Berkeley Parser is very similar to that corresponding to Stanford Parser despite a 4-point difference in F1 score between the two parsers. Another example is that Charniak parser performs slightly worse than the other three on MT task, though it achieves the 2nd best parsing performance in all the parsers. This interesting finding shows that the performance of syntax-based MT systems is not very sensitive to the parsing performance of the parsers used in rule extraction.

6.4 Preferences for Parsers

We also investigate the preferences for different parsers in our system. Table 6 shows the weights of the parser indicator features learned by MERT, as well as the number of edges generated by applying the rules corresponding to different parsers during decoding. Both of the metrics are used to evaluate the contributions of the parsers to MT decoding. We see that though Stanford Parser and Berkeley Parser are shown to be relatively more preferred by the decoder, there are actually no significant differences in the degrees of the contributions of different parsers. This result also confirms the fact observed in Table 5 that the MT system does not have special preferences for different parsers.

Indicator	Weight	# of edges (Dev.)	# of edges (MT04)	# of edges (MT05)
Stanford	0.1990	7.7 M	169.2 M	101.7 M
Berkeley	0.1982	7.7 M	166.3 M	100.2 M
Collins	0.1690	6.9 M	149.9 M	93.1 M
Charniak	0.1729	7.1 M	156.5 M	97.2 M

Table 6: Preferences for different parsers.

Though Table 6 provides some information about the contributions of different parsers, it still does not answer how often these rules are really used to generate final (1-best) translation. Table 7 gives an answer to this question. We see that, following the similar trend in Table 5, different parsers have nearly equal contributions in generating final translation.

Indicator	# of rules used in 1-best (Dev.)	# of rules used in 1-best (MT04)	# of rules used in 1-best (MT05)
Stanford	2,410	23,513	14,357
Berkeley	2,455	23,878	14,670
Collins	2,309	22,654	13,815
Charniak	2,269	22,406	13,731

Table 7: Numbers of rules used in generating final (1-best) translation.

6.5 Rule Extraction with *k*-best Parses

We also conduct experiments to compare the effectiveness of multi-parser based rule extraction and rule extraction with *k*-best parses generated by a single parser. As Berkeley parser is one of the best-performing parsers in previous experiments, we employ it to generate *k*-best parses in this set of experiment. As shown in Figure 3, both of the methods improve the BLEU scores by enlarging the set of parse trees used in rule extraction. Compared to *k*-best extraction, multi-parser extraction shows consiste-

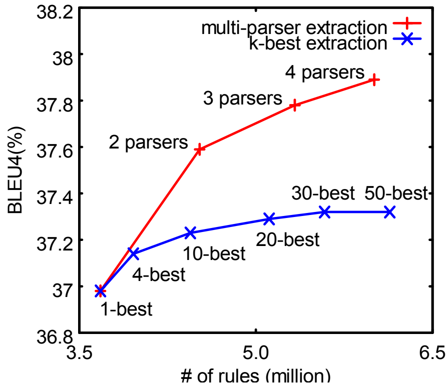


Figure 3: Multi-parser based rule extraction vs. rule extraction with *k*-best parses (MT05).

ntly better BLEU scores. Using 4 different parsers, it achieves an improvement of 0.6 BLEU points over *k*-best extraction where even 50-best parses are used.

Finally, we extend multi-parser based rule extraction to extracting rules from the *k*-best parses generated by multiple parsers. Figure 4 shows the results on "S + B + Co + Ch" system. We see that multi-parser based rule extraction can benefit from *k*-best parses, and yields a modest (+0.2 BLEU points) improvement when extracting from 10-best parses. However, since *k*-best extraction generally results in much slower extraction speed, it might not be a good choice to use *k*-best parses to improve our method in practice.

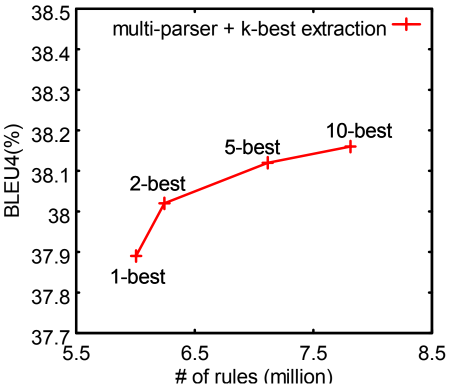


Figure 4: Multi-parser based rule extraction & rule extraction with *k*-best parses (MT05).

7 Discussion and Future Work

In this work, all the parsers are trained using the same treebank. To obtain diversified parse trees for multi-parser based rule extraction, an alternative way is to learn parsers on treebanks annotated by different organizations (e.g. Penn Treebank and ICE-GB corpus). Since different treebanks can provide us with more diversity in parsing, we believe that our system can benefit a lot from the parsers that are learned on multiple different treebanks individually. But here is a problem that due to the different annotation standards used, there is generally an incompatibility between treebanks annotated by different organizations. It will result in that we cannot straightforwardly mix the resulting rule sets (or *heterogeneous grammars* for short) for probability estimation as well as the use for decoding. To solve this problem, a simple solution might be that we transform the incompatible rules into a unified form. Alternatively, we can use *heterogeneous decoding* (or *parsing*) techniques (Zhu et al., 2010) to make use of heterogeneous grammars in the stage of decoding. Both topics are very interesting and worth studying in our future work.

Besides *k*-best extraction, our method can also be applied to other rule extraction schemes, such as forest-based rule extraction. As (Mi and Huang, 2008) has shown that forest-based extraction is more effective than *k*-best extraction in improving translation accuracy, it is expected to achieve further improvements by using multi-parser based rule extraction and forest-based rule extraction together.

8 Conclusions

In this paper, we present a simple and effective method to improve rule coverage by using multiple parsers in translation rule extraction. Experimental results show that

- Using multiple parsers in rule extraction achieves large improvements of rule coverage over the baseline method where only a single parser is used, as well as a +0.9 BLEU improvement on both NIST 2004 and 2005 test corpora.
- The MT system can be further improved by using multiple parsers and *k*-best parses together. However, with the consideration of extraction speed, it might not be a good choice to use *k*-best parses to improve multi-parser based rule extraction in practice.
- The MT performance is not influenced by the parsing performance of the parsers used in rule extraction very much. Actually, the MT system does not show different preferences for different parsers.

Acknowledgements

This work was supported in part by the National Science Foundation of China (60873091) and the Fundamental Research Funds for the Central Universities (N090604008). The authors would like to thank the anonymous reviewers and Tongran Liu for their pertinent comments for improving the early version of this paper, and Rushan Chen for building parts of the baseline system.

References

Brooke Cowan, Ivona Kučerová and Michael Collins. 2006. A discriminative model for tree-to-tree translation. In *Proc. of* EMNLP 2006, pages 232-241.

Steve DeNeefe, Kevin Knight, Wei Wang and Daniel Marcu. 2007. What Can Syntax-based MT Learn from Phrase-based MT? In *Proc. of* EMNLP 2007, pages 755-763.

Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proc. of* ACL 2005, Ann Arbor, Michigan, pages 541-548.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proc. of* ACL 2003, pages 205-208.

Michel Galley, Mark Hopkins, Kevin Knight and Daniel Marcu. 2004. What's in a translation rule? In *Proc. of* HLT-NAACL 2004, Boston, USA, pages 273-280.

Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang and Ignacio Thayer. 2006. Scalable inferences and training of context-rich syntax translation models. In *Proc. of* COLING/ACL 2006, Sydney, Australia, pages 961-968.

Liang Huang and David Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proc. of* ACL 2007, Prague, Czech Republic, pages 144-151.

Liang Huang, Kevin Knight and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proc. of* AMTA 2006, pages 66-73.

Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proc. of* EMNLP 2004, Barcelona, Spain, pages 388-395.

Yang Liu, Qun Liu and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proc. of* COLING/ACL 2006, Sydney, Australia, pages 609-616.

Yang Liu, Yajuan Lü and Qun Liu. 2009. Improving Tree-to-Tree Translation with Packed Forest. In *Proc. of* ACL 2009, pages 558-566.

Daniel Marcu, Wei Wang, Abdessamad Echihabi and Kevin Knight. 2006. SPMT: Statistical machine translation with syntactified target language phrases. In *Proc. of* EMNLP 2006, Sydney, Australia, pages 44-52.

Haitao Mi and Liang Huang. 2008. Forest-based Translation Rule Extraction. In *Proc. of* EMNLP 2008, pages 206-214.

Chris Quirk, Arul Menezes and Colin Cherry. 2005. Dependency treelet translation: Syntactically informed phrasal SMT. In *Proc. of* ACL 2005, pages 271-279.

Libin Shen, Jinxi Xu and Ralph Weischedel. 2008. A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proc. of* ACL/HLT 2008, pages 577-585.

Ashish Venugopal, Andreas Zollmann, Noah A. Smith and Stephan Vogel. 2008. Wider Pipelines: K-best Alignments and Parses in MT Training. In *Proc. of* AMTA 2008, pages 192-201.

Wei Wang, Kevin Knight and Daniel Marcu. 2007. Binarizing Syntax Trees to Improve Syntax-Based Machine Translation Accuracy. In *Proc. of* EMNLP-CoNLL 2007, Prague, Czech Republic, pages 746-754.

Tong Xiao, Mu Li, Dongdong Zhang, Jingbo Zhu and Ming Zhou. 2009. Better Synchronous Binarization for Machine Translation. In *Proc. of* EMNLP 2009, Singapore, pages 362-370.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical machine translation model. In *Proc. of* ACL 2001, pages 132-139.

Hao Zhang, Liang Huang, Daniel Gildea and Kevin Knight. 2006. Synchronous Binarization for Machine Translation. In *Proc. of* HLT-NAACL 2006, New York, USA, pages 256- 263.

Min Zhang, Hongfei Jiang, Aiti Aw, Haizhou Li, Chew Lim Tan and Sheng Li. 2008. A Tree Sequence Alignment-based Tree-to-Tree Translation Model. In *Proc. of* ACL/HLT 2008, pages 559-567.

Muhua Zhu, Jingbo Zhu and Tong Xiao. 2010. Heterogeneous Parsing via Collaborative Decoding. In *Proc. of* COLING 2010.