Filtering Syntactic Constraints for Statistical Machine Translation

Hailong Cao and **Eiichiro Sumita**
Language Translation Group, MASTAR Project
National Institute of Information and Communications Technology
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan, 619-0289
{hlcao, eiichiro.sumita }@nict.go.jp

Abstract

Source language parse trees offer very useful but imperfect reordering constraints for statistical machine translation. A lot of effort has been made for soft applications of syntactic constraints. We alternatively propose the selective use of syntactic constraints. A classifier is built automatically to decide whether a node in the parse trees should be used as a reordering constraint or not. Using this information yields a 0.8 BLEU point improvement over a full constraint-based system.

1 Introduction

In statistical machine translation (SMT), the search problem is NP-hard if arbitrary reordering is allowed (Knight, 1999). Therefore, we need to restrict the possible reordering in an appropriate way for both efficiency and translation quality. The most widely used reordering constraints are IBM constraints (Berger et al., 1996), ITG constraints (Wu, 1995) and syntactic constraints (Yamada et al., 2000; Galley et al., 2004; Liu et al., 2006; Marcu et al., 2006; Zollmann and Venugopal 2006; and numerous others). Syntactic constraints can be imposed from the source side or target side. This work will focus on syntactic constraints from source parse trees.

Linguistic parse trees can provide very useful reordering constraints for SMT. However, they are far from perfect because of both parsing errors and the crossing of the constituents and formal phrases extracted from parallel training data. The key challenge is how to take advantage of the prior knowledge in the linguistic parse trees without affecting the strengths of formal phrases. Recent efforts attack this problem by using the constraints softly (Cherry, 2008; Marton and Resnik, 2008). In their methods, a candidate translation gets an extra credit if it respects the parse tree but may incur a cost if it violates a constituent boundary.

In this paper, we address this challenge from a less explored direction. Rather than use all constraints offered by the parse trees, we propose using them selectively. Based on parallel training data, a classifier is built automatically to decide whether a node in the parse trees should be used as a reordering constraint or not. As a result, we obtain a 0.8 BLEU point improvement over a full constraint-based system.

2 Reordering Constraints from Source Parse Trees

In this section we briefly review a constraint-based system named IST-ITG (Imposing Source Tree on Inversion Transduction Grammar, Yamamoto et al., 2008) upon which this work builds.

When using ITG constraints during decoding, the source-side parse tree structure is not considered. The reordering process can be more tightly constrained if constraints from the source parse tree are integrated with the ITG constraints. IST-ITG constraints directly apply source sentence tree structure to generate the target with the following constraint: the target sentence is obtained by rotating any node of the source sentence tree structure.

After parsing the source sentence, a bracketed sentence is obtained by removing the node syntactic labels; this bracketed sentence can then be directly expressed as a tree structure. For example[1], the parse tree "(S1 (S (NP (DT This)) (VP (AUX is) (NP (DT a) (NN pen)))))" is obtained from the source sentence "This is a pen", which consists of four words. By removing

[1] We use English examples for the sake of readability.

Proceedings of the ACL 2010 Conference Short Papers, pages 17–21,
Uppsala, Sweden, 11-16 July 2010. ©2010 Association for Computational Linguistics

the node syntactic labels, the bracketed sentence "((This) ((is) ((a) (pen))))" is obtained. Such a bracketed sentence can be used to produce constraints.

For example, for the source-side bracketed tree "((f1 f2) (f3 f4)) ", eight target sequences [e1, e2, e3, e4], [e2, e1, e3, e4], [e1, e2, e4, e3], [e2, e1, e4, e3], [e3, e4, e1, e2], [e3, e4, e2, e1], [e4, e3, e1, e2], and [e4, e3, e2, e1] are possible. For the source-side bracketed tree "(((f1f2) f3) f4)," eight sequences [e1, e2, e3, e4], [e2, e1, e3, e4], [e3, e1, e2, e4], [e3, e2, e1, e4], [e4, e1, e2, e3], [e4, e2, e1, e3], [e4, e3, e1, e2], and [e4, e3, e2, e1] are possible. When the source sentence tree structure is a binary tree, the number of word orderings is reduced to 2^{N-1} where N is the length of the source sentence.

The parsing results sometimes do not produce binary trees. In this case, some subtrees have more than two child nodes. For a non-binary subtree, any reordering of child nodes is allowed. For example, if a subtree has three child nodes, six reorderings of the nodes are possible.

3 Learning to Classify Parse Tree Nodes

In IST-ITG and many other methods which use syntactic constraints, all of the nodes in the parse trees are utilized. Though many nodes in the parse trees are useful, we would argue that some nodes are not trustworthy. For example, if we constrain the translation of "f1 f2 f3 f4" with node N2 illustrated in Figure 1, then word "e1" will never be put in the middle the other three words. If we want to obtain the translation "e2 e1 e4 e3", node N3 can offer a good constraint while node N2 should be filtered out. In real corpora, cases such as node N2 are frequent enough to be noticeable (see Fox (2002) or section 4.1 in this paper).

Therefore, we use the definitions in Galley et al. (2004) to classify the nodes in parse trees into two types: frontier nodes and interior nodes. Though the definitions were originally made for target language parse trees, they can be straightforwardly applied to the source side. A node which satisfies both of the following two conditions is referred as a frontier node:

- All the words covered by the node can be translated separately. That is to say, these words do not share a translation with any word outside the coverage of the node.
- All the words covered by the node remain contiguous after translation.

Otherwise the node is an interior node.

For example, in Figure 1, both node N1 and node N3 are frontier nodes. Node N2 is an interior node because the source words f2, f3 and f4 are translated into e2, e3 and e4, which are not contiguous in the target side.

Clearly, only frontier nodes should be used as reordering constraints while interior nodes are not suitable for this. However, little work has been done on how to explicitly distinguish these two kinds of nodes in the source parse trees. In this section, we will explore building a classifier which can label the nodes in the parse trees as frontier nodes or interior nodes.

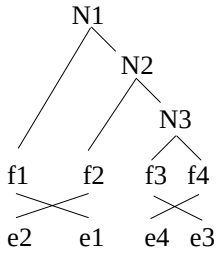

Figure 1: An example parse tree and alignments

3.1 Training

Ideally, we would have a human-annotated corpus in which each sentence is parsed and each node in the parse trees is labeled as a frontier node or an interior node. But such a target language specific corpus is hard to come by, and never in the quantity we would like.

Instead, we generate such a corpus automatically. We begin with a parallel corpus which will be used to train our SMT model. In our case, it is the FBIS Chinese-English corpus.

Firstly, 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.

Secondly, we use GIZA++ to align the sentences in both the Chinese-English and English-Chinese directions. We combine the alignments using the "grow-diag-final-and" procedure provided with MOSES (Koehn, 2007). Because there are many errors in the alignment, we remove the links if the alignment count is less than three for the source or the target word. Additionally, we also remove notoriously bad links in

{de, le} × {the, a, an} following Fossum and Knight (2008).

Thirdly, given the parse trees and the alignment information, we label each node as a frontier node or an interior node according to the definition introduced in this section. Using the labeled nodes as training data, we can build a classifier. In theory, a broad class of machine learning tools can be used; however, due to the scale of the task (see section 4), we utilize the Pegasos [2] which is a very fast *SVM* solver (Shalev-Shwartz et al, 2007).

3.2 Features

For each node in the parse trees, we use the following feature templates:

- A context-free grammar rule which rewrites the current node (In this and all the following grammar based features, a mark is used to indicate which non terminal is the current node.)
- A context-free grammar rule which rewrites the current node's father
- The combination of the above two rules
- A lexicalized context-free grammar rule which rewrites the current node
- A lexicalized context-free grammar rule which rewrites the current node's father
- Syntactic label, head word, and head POS tag of the current node
- Syntactic label, head word, and head POS tag of the current node's left child
- Syntactic label, head word, and head POS tag of the current node's right child
- Syntactic label, head word, and head POS tag of the current node's left brother
- Syntactic label, head word, and head POS tag of the current node's right brother
- Syntactic label, head word, and head POS tag of the current node's father
- The leftmost word covered by the current node and the word before it
- The rightmost word covered by the current node and the word after it

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 lexical reordering model was used in our experiments.

The translation model was created from the FBIS corpus. We used a 5-gram language model trained with modified Knesser-Ney smoothing. The language model was trained on the target side of FBIS corpus and the Xinhua news in GIGAWORD corpus. The development and test sets are from NIST MT08 evaluation campaign. Table 1 shows the statistics of the corpora used in our experiments.

Data	Sentences	Chinese words	English words
Training set	243,698	7,933,133	10,343,140
Development set	1664	38,779	46,387
Test set	1357	32377	42,444
GIGAWORD	19,049,757	-	306,221,306

Table 1: Corpora statistics

4.1 Experiments on Nodes Classification

We extracted about 3.9 million example nodes from the training data, i.e. the FBIS corpus. There were 2.37 million frontier nodes and 1.59 million interior nodes in these examples, give rise to about 4.4 million features. To test the performance of our classifier, we simply use the last ten thousand examples as a test set, and the rest being used as Pegasos training data. All the parameters in Pegasos were set as default values. In this way, the accuracy of the classifier was 71.59%.

Then we retrained our classifier by using all of the examples. The nodes in the automatically parsed NIST MT08 test set were labeled by the classifier. As a result, 17,240 nodes were labeled as frontier nodes and 5,736 nodes were labeled as interior nodes.

4.2 Experiments on Chinese-English SMT

In order to confirm that it is advantageous to distinguish between frontier nodes and interior nodes, we performed four translation experiments.

The first one was a typical beam search decoding without any syntactic constraints.

All the other three experiments were based on the IST-ITG method which makes use of syntac-

[2] http://www.cs.huji.ac.il/~shais/code/index.html

tic constraints. The difference between these three experiments lies in what constraints are used. In detail, the second one used all nodes recognized by the parser; the third one only used frontier nodes labeled by the classifier; the fourth one only used interior nodes labeled by the classifier.

With the exception of the above differences, all the other settings were the same in the four experiments. Table 2 summarizes the SMT performance.

Syntactic Constraints	BLEU
none	17.26
all nodes	16.83
frontier nodes	17.63
interior nodes	16.59

Table 2: Comparison of different constraints by SMT quality

Clearly, we obtain the best performance if we constrain the search with only frontier nodes. Using just frontier yields a 0.8 BLEU point improvement over the baseline constraint-based system which uses all the constraints.

On the other hand, constraints from interior nodes result in the worst performance. This comparison shows it is necessary to explicitly distinguish nodes in the source parse trees when they are used as reordering constraints.

The improvement over the system without constraints is only modest. It may be too coarse to use pare trees as hard constraints. We believe a greater improvement can be expected if we apply our idea to finer-grained approaches that use constraints softly (Marton and Resnik (2008) and Cherry (2008)).

5 Conclusion and Future Work

We propose a selectively approach to syntactic constraints during decoding. A classifier is built automatically to decide whether a node in the parse trees should be used as a reordering constraint or not. Preliminary results show that it is not only advantageous but necessary to explicitly distinguish between frontier nodes and interior nodes.

The idea of selecting syntactic constraints is compatible with the idea of using constraints softly; we plan to combine the two ideas and obtain further improvements in future work.

Acknowledgments

We would like to thank Taro Watanabe and Andrew Finch for insightful discussions. We also would like to thank the anonymous reviewers for their constructive comments.

Reference

A.L. Berger, P.F. Brown, S.A.D. Pietra, V.J.D. Pietra, J.R. Gillett, A.S. Kehler, and R.L. Mercer. 1996. Language translation apparatus and method of using context-based translation models. United States patent, patent number 5510981, April.

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

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

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

Victoria Fossum and Kevin Knight. 2008. Using bilingual Chinese-English word alignments to resolve PP attachment ambiguity in English. In *AMTA Student Workshop*.

Heidi J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *EMNLP*.

Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *HLT-NAACL*.

Kevin Knight. 1999. Decoding complexity in word replacement translation models. *Computational Linguistics*, 25(4):607–615.

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*.

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*.

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

Shai Shalev-Shwartz, Yoram Singer and Nathan Srebro. 2007. Pegasos: Primal estimated sub-gradient solver for SVM. In *ICML*.

Dekai Wu. 1995. Stochastic inversion transduction grammars with application to segmentation, bracketing, and alignment of parallel corpora. In *IJCAI*.

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

Hirofumi Yamamoto, Hideo Okuma and Eiichiro Sumita. 2008. Imposing constraints from the source tree on ITG constraints for SMT. In *Workshop on syntax and structure in statistical translation*.

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