A Source Dependency Model for Statistical Machine Translation

Deyi Xiong, Min Zhang, Aiti Aw and Haizhou Li

Human Language Technology

Institute for Infocomm Research 1 Fusionopolis Way, #21-01 Connexis, Singapore 138632. {dyxiong, mzhang, aaiti, hli}@i2r.a-star.edu.sg

Abstract

In the formally syntax-based MT, a hierarchical tree generated by synchronous CFG rules associates the source sentence with the target sentence. In this paper, we propose a source dependency model to estimate the probability of the hierarchical tree generated in decoding. We develop this source dependency model from word-aligned corpus, without using any linguistically motivated parsing. Our experimental results show that integrating the source dependency model into the formally syntax-based machine translation significantly improves the performance on Chinese-to-English translation tasks.

1 Introduction

In the word-based translation model introduced by IBM (Brown et al., 1993), the hidden variable, *word alignment*, associates the source sentence (*c*) with the target sentence (*e*). This model has been advanced to the phrase-based and syntax-based models, and the hidden variable has also been transformed into new forms: *phrase alignment* in phrase-based translation and *hierarchical tree* in syntax-based translation. In all cases, we can search the best translation among all possible target sentences and hidden variables through a log-linear model (Och and Ney, 2002)

$$\hat{\mathbf{e}} = \operatorname*{argmax}_{e,h}(\Sigma_i \lambda_i f_i(c,h,e)) \tag{1}$$

where f_i are feature functions which are dependent not only on c and e but also on the hidden variable h. In this paper, we introduce a new feature function f(c, h, e) = Pr(h) to model h in the formally syntax-based machine translation¹, where h is a hierarchical tree shared by c and e. Probabilistic formal grammars are not adequate to model h because they suffer from the problem of overgeneralization: producing many different hierarchical trees with the same probability. This is the reason why we incorporate an independent model Pr(h).

Since each node in the hierarchical tree contains the source and target part, we can project the tree onto either the source side or the target side. Pr(h)is therefore factored into

$$Pr(h) \approx Pr(h_c)Pr(h_e)$$
 (2)

where h_c and h_e are the tree projections of h on the source and target side respectively.

Without loss of generality, we discuss Pr(h) within the context of Bracketing Transduction Grammar (BTG) (Wu, 1997), which is a binary synchronous CFG widely adopted in machine translation. We focus on $Pr(h_c)$ in this paper and leave the modeling of $Pr(h_e)$ to be our future work. To calculate $Pr(h_c)$, we use a source dependency model.

The challenge to our source dependency model is the acquisition of training data: source dependency trees. The source dependency tree used for training has to satisfy two conditions: 1) it is not necessarily linguistically sensible; and 2) it is produced under bilingual context. We observe that we can obtain such dependency trees from word alignments because word alignments implicitly contain hierarchical structures of both source and target language.

¹We inherit the definition of the formally syntax-based MT from (Chiang, 2005). It refers to syntax-based MT which uses synchronous CFG with no linguistic commitment.

To uncover the hidden structures from word alignments, we adopt the algorithm of (Zhang et al., 2008). The algorithm decomposes word-aligned sentence pairs into hierarchical trees where leaf nodes are permuted from left to right according to the target language word order. From these trees, we extract the normalized source trees. In order to obtain dependency relations of source words, we annotate nodes with head words and labels. Without introducing conflict with the original definition of dependency tree, we call the annotated tree obtained from the above steps source dependency tree. Using these source dependency trees, we develop our source dependency model.

To the best of our knowledge, this is the first attempt to use a source dependency model in machine translation, in which the model is built from word alignments without using any linguistic parsing.

The rest of the paper is organized as follows. In section 2, we learn source dependency trees from word alignments. At first we briefly introduce (Zhang et al., 2008)'s hierarchical analysis of word alignment, and then describe the way to transform the bilingual hierarchical analysis into source dependency tree with a concrete example. In section 3, we develop our source dependency model. In section 4, we describe new decoding with a source dependency model. In section 5, we present our experiments. In section 6, we discuss our method and lay out some future directions. Section 7 presents the discussion of related work and section 8 concludes the paper.

2 Acquisition of Source Dependency Trees from Word Alignments

In this section, we use (Zhang et al., 2008)'s shiftreduce algorithm (SRA) to decompose word alignments into hierarchical trees. Since our focus is not hierarchical analysis of word alignments, we ignore the description of the algorithm. Our interest here is to transform the hierarchical analysis of word alignment into a source dependency tree.

2.1 Hierarchical Analysis of Word Alignment

Given an arbitrary word-level alignment as an input, SRA is able to output a tree representation of the word alignment (a.k.a **decomposition tree**). Each node of the tree is a *tight* phrase, where all boundary words are aligned.

Figure 1a shows an example of many-to-many alignment, where the source language is Chinese and the target language is English. Each word is indexed with its occurring position from left to right. Figure 2b is the tree representation of the word alignment after hierarchical analysis using SRA. We use [s-t|u-v] to denote a bilingual **item**, where s, t and u, v are the beginning and ending index in the target and source sentence, respectively. Note that a bilingual item is not equivalent to a bilingual phrase. The words within [s-t] of a bilingual item may be aligned to words outside [u-v] or vise versa. Each node of the tree is composed of a series of items, where all items together form a bilingual phrase. If an item can be decomposed into smaller items, it is said to be decomposable otherwise nondecomposable. For example, in Figure 1b, item [1-13]1-11], [3-6]8-11] and [7-9]4-5] are decomposable while [2|2-3] and [10|3] are non-decomposable.

All leaf nodes are composed of nondecomposable items. But nonterminal nodes also frequently include non-decomposable items. There are two reasons for this. First, unaligned target words are attached as high as possible in SRA.² This means that they are included in nonterminal nodes as a non-decomposable item rather than an independent leaf node. Second. target words whose counterparts on the source side overlap (due to many-to-many alignment links) will be non-decomposable items in non-terminal nodes. For example, [2|2-3] and [10|3] are such non-decomposable items.

2.2 Source Dependency Tree Transformation

While the decomposition tree of word alignment shows the hierarchical structure of source and target sentence, dependency relations among source words are not directly visible from the tree. We need to transform the decomposition tree into a source dependency tree, where dependency relations are explicit. We take three steps to complete this task: normalization, binarization and finally annotation.

²Note that unaligned source words are not included in our decomposition trees.



Figure 1: Illustration of acquisition of source dependency tree from word alignment.

2.2.1 Normalization

There are various types of links (many-toone, one-to-many and many-to-many) in word alignments, some of which introduce noises into decomposition trees, including notorious nondecomposable items in nonterminal nodes. The goal of normalization is to deal with these noises and clean the decomposition trees as much as possible. We define 3 types of operations to normalize a decomposition tree as follows

• Merge If a source word is aligned to multiple consecutive target words, each target word will form a leaf node with the same source word in the decomposition tree. In Figure 1b, leaf nodes ([3|8]) and ([4|8]), ([9|5]) and ([8|5]) are formed under such situation. The operator

Merge combines all such leaf nodes into one leaf node, and correspondingly deletes the previous parent node.

- **Decompose** If a target word is aligned to multiple consecutive source words, only one leaf node will be formed for all the source words aligned to the target word in the decomposition tree. The Decompose operator, contrary to the Merge operator, decomposes this leaf node into multiple leaf nodes. Each source word will form a leaf node together with the target word. A preterminal node is generated to dominate all new leaf nodes.
- Delete Unaligned target words or target words involved in many-to-many alignments form non-decomposable items in non-terminal nodes of the decomposition tree. Currently we simply delete all such non-decomposable items. This deletion will remove some source words from our final dependency trees. For example, in Figure 2b, item [2|2-3] and [10|3] are deleted after normalization, which correspondingly removes the source word [2] and [3]. To deal with this problem, we remove less probable links in word alignments before hierarchical analysis. A less probable link connects a source word C to a target word E with a relatively low probability of Pr(C|E) or Pr(E|C). These links can be considered as noises, introducing manyto-many alignments and overlaps. In Figure 1a, the link between the source word [3] and the target word [10] is a less probable link. Removing this link, we can keep the source word [2] and [3] after normalization. However, removing less probable links is not risk-free since changes in word alignments will change the decomposition trees. We will discuss this further in our experiments.

After carrying out all possible operations listed above on the decomposition tree, we only keep source words in each node. Figure 1c shows the result after we normalize the tree in Figure 1b.

2.2.2 Binarization

Since we use binary grammars (BTGs), we binarize a tree node whose fan-out is larger than 2 after normalization. There are two simple and popular binarization methods: left-binarization and **right-binarization**. Given a node n that dominates k children $n_1, ..., n_k$ where k > 2. The leftbinarization, first combines n_1 and n_2 and generates a new node n_{12} to dominate them. The new node n_{12} will replace n_1 and n_2 in the children sequence: n_{12}, n_3, \dots, n_k . Then the node n_{12} and n_3 are combined and a new node n_{123} is generated. The procedure repeats until the last child node n_k is combined with the newly generated node $n_{1...k-1}$ and a new node $n_{1...k}$ is generated to replace n. The rightbinarization follows the similar procedure but runs from right to left on the children sequence. Figure 1d shows the binary tree of Figure 1c after the leftbinarization is used.

Our preliminary results show that the leftbinarization is better than the right-binarization. Therefore we use left-binarization in all our experiments.

2.2.3 Annotation

The annotation is the final step for transforming a decomposition tree of word alignment into a source dependency tree. In this step, we attach two annotation elements for each node: label and lexical item (i.e. head word and head tag). To obtain annotations, we tag the source language using an external part-of-speech (POS) tagger. Figure 1e shows the final annotated tree, where l:hw/ht in each node denote the label, head word and POS tag of the head word, respectively.

- Label For each leaf node, we annotate it with its POS tag. For example, the leaf node ([8]) and ([9]) are labeled with VV and JJ respectively. For a binary nonterminal node, we use only two labels: X and Y. The former denotes a straight order between two children nodes while the latter an inverted order. For instance, the highlighted node (in bold) in Figure 1e is labeled with Y. The left child of this node covers the source sequence [8-11] while the right child covers [4-5]. The two children are in reverse order.
- Lexical Item To obtain dependency relations, we annotate head word and head tag for each

POS Tags	Weight
VA, VC, VE, VV	8
AD, AS, P, BA, SB, LB, LC, ETC,	7
DER, DEV	
FW, NN, PN	6
NR, NT	5
M, DT, JJ	4
OD, CD	3
DEC, DEG, CS	2
CC, IJ, MSP, PU, SP, ON	1

Table 1: Predefined weights for all POS tags. The tags listed here are from the Chinese Penn Treebank (Xue and Xia, 2000).

node. The head word of a parent node is derived from the head word of its head child. To determine which child node is the head child, we use heuristic rules based on predefined weights for all POS tags (shown in table 1). For any two children nodes n_l and n_r with head tag p and q respectively, if the weight of p is larger than that of q, n_l will be the head child; otherwise n_r . If the weights of p and qare equal, we select the right child (n_r) as the head child.

3 Source Dependency Model

Since our dependency trees are binary, each nonterminal node has only one dependency relation between the head child and modifier child. We calculate $Pr(h_c)$ as follows

$$Pr(h_c) = \prod_{i=1}^{N} Pr(r_i)$$
(3)

where N is the number of nonterminal nodes in h_c , r is the dependency relation between the head child and modifier child of the *i*th nonterminal node.

Let n_p be the parent node, n_h be the head child node and n_m be the modifier child node. Similar to (Collins, 1999), we factor Pr(r) into two distributions. The first distribution is the probability of generating the label and head tag of the modifier node given context features from the parent node and head node:

$$Pr(n_m.l, n_m.ht|n_p.l, n_h.l, n_h.ht, n_h.hw, dir) \quad (4)$$

L	$Pr(n_m.l, n_m.ht \cdot)$	$Pr(n_m.hw \cdot)$
0	All	All
1	$n_p.l, n_h.l, n_h.ht, dir$	$n_p.l, n_h.l, n_h.ht, n_h.hw$
2	$n_p.l, n_h.l, n_h.ht$	$n_h.ht, n_h.hw$
3	$n_h.l, n_h.ht$	$n_h.ht$

Table 2: Back-off structures for the two probabilities. L means Back-off level.

where dir is the direction of n_m relative to the head node n_h . In the second distribution, the head word of n_m is generated with the probability:

$$Pr(n_m.hw|n_p.l, n_h.l, n_h.ht, n_h.hw, n_m.l, n_m.ht, dir)$$
(5)

We obtain maximum-likelihood estimates of the two distributions (formula (4) and (5)) using frequencies gathered from source dependency trees which we produce from word-aligned corpus. To deal with the data sparseness problem, we smooth the two probabilities through Witten-Bell interpolation, similar to (Collins, 1999). Table 2 shows the back-off structures for the smoothing. Further, for words occurring less than 5 times in training data, and words in test data which have never been seen in training, we replace them with the "UNKNOWN" token.

As described in the introduction, $Pr(h_c)$ is integrated into the log-linear translation model as a new feature. The weight of this new feature, like the weights of other features, is tuned via minimumerror-rate training (MERT) (Och, 2003) on a development set.

4 Decoding with Source Dependency Model

Since we use a bottom-up CKY parsing algorithm for decoding, computing the source dependency model score is quite straightforward. When a new node of the hierarchical tree h_c is being constructed, we use the predefined POS tag weight table (Table 1) to determine the head. If the BTG lexical rule $(A \rightarrow x/y)$ is used to produce a leaf node upon a source span, we select the rightmost source word with the highest POS tag weight within the source span as the head word for the leaf node. If the BTG merging rules $(A \rightarrow [A, A]|\langle A, A \rangle)$ are used to combine two neighboring nodes and form a parent node dominating them, we determine the head node according to the weights of their head tags. The head word of the selected node will be propagated to the parent node. Once the head is determined, the decoder can quickly collect the context features of formula (4) and (5).

Here we should note two things. First, the head word is forcedly assigned according to POS tag weights. No probability is involved in this decision. Second, we do not add the head word to the state of hypothesis. This means we do not expand the search space which the original decoder has to explore. These two simplifications greatly reduce potential workload caused by introducing a dependency model into decoding.

5 Experiments

We carried out experiments to examine the effect of the source dependency model on Chinese-to-English translation tasks.

5.1 Experimental Setup

Our baseline is a formally syntax-based system using BTG, developed by following (Xiong et al., 2006). The training data is from FBIS corpus, which contains 6.59M Chinese words and 8.04M English words. To obtain word-level alignments, we ran GIZA++ (Och and Ney, 2000) on the corpus in both directions, and applied the "grow-diag-final" refinement rule (Koehn et al., 2005) to produce the final many-to-many word alignments. We trained a fourgram language model using Xinhua section of the English Gigaword corpus (181.1M words) with the SRILM toolkit (Stolcke, 2002).

For the efficiency of MERT, we built our development set (580 sentences) using sentences not exceeding 50 characters from the NIST MT-02 set. We evaluated our systems on the NIST MT-05 and MT-03 test sets using case-sensitive BLEU-4. Statistical significance in BLEU score differences was assessed by paired bootstrap re-sampling (Koehn, 2004).

5.2 Source Dependency Model Training

Because SRA runs very quickly, which produces within a few minutes all decomposition trees from our word-aligned sentence pairs, we can easily train various source dependency models following the steps described in Section 2. In order to allow the normalization step to keep more source words, we removed less probable links in word alignments. First, we calculate lexicon translation probabilities in both directions (Pr(C|E) and Pr(E|C)) from the original word alignments. Then we search on the source side or the target side for words which are involved in oneto-many alignments as follows

- Removing on the source side If a source word C is aligned to multiple target words, we find the link with the highest probability Pr(E|C), which connects to the target word E. For any other link connecting C with target word E', if Pr(E'|C) < t × Pr(E|C), we remove this link. We call this removing method c_t, where t is the removing threshold.
- Removing on the target side If a target word E is aligned to multiple source words, we remove any link (connecting to the source word C') whose probability Pr(C'|E) is less than t × Pr(C|E), where Pr(C|E) is the highest probability among all links emitting from E. We refer to this removing method as e_t.

We set the threshold $t = \{0.05, 0.1, 0.2, 0.4\}$ for both c_t and e_t . Table 3 shows the statistics about the source dependency model (SDM) training using c_t and e_t . We have the following observations

- Excluding unary nodes, most of the remaining nodes are binary nodes, which account for more than 93%. This is in line with (Zhang et al., 2008)'s finding. It implies that binary synchronous grammars, like BTG, are adequate for machine translation.
- Wider thresholds lead to more source words, nodes and dependency relations. This is help-ful for our source dependency model training, since we have more training instances. On the other hand, however, it could also mislead the training because the increased dependency relations may be wrong as word alignments are changed.
- On the same threshold level, c_t produces more training instances than e_t does.

	#SW	#BN	#NAN	#DR
$1/\infty$	4.19M	2.86M	0.20M	0.94M
$c_{0.05}$	4.39M	2.99M	0.21M	0.98M
$c_{0.1}$	4.48M	3.06M	0.21M	1.01M
$c_{0.2}$	4.59M	3.16M	0.21M	1.04M
$c_{0.4}$	4.73M	3.28M	0.21M	1.09M
$e_{0.05}$	4.34M	2.97M	0.21M	0.96M
$e_{0.1}$	4.39M	3.01M	0.21M	0.98M
$e_{0.2}$	4.46M	3.07M	0.21M	0.99M
$e_{0.4}$	4.52M	3.13M	0.21M	1.01M

Table 3: Statistics on the source dependency model training. #SW, #BN and #NAN denote the number of source words kept, binary nodes and *n*-ary nodes where n > 2 after normalization, respectively. #DR is the number of dependency relations extracted from source dependency trees. $1/\infty$ denotes that no less probable links are removed.

5.3 Results

Table 4 summarizes the results of our experiments on Chinese-to-English translation. These results confirm that the source dependency model can significantly improve performance as measured by the BLEU score, with a consistent pattern of results across the MT-03 and MT-05 test sets. Without removing less probable links (denoted as $1/\infty$), SDM still gains significant improvements over the baseline. Removing less probable links at t = 0.05, the performance drops marginally when compared with $1/\infty$ in most cases. Lifting the threshold to 0.1 and 0.2, we obtain the highest improvement over the baseline (0.7 on MT-03 and 0.8 on MT-05), suggesting that the source dependency model benefits from the increase of training instances. Larger thresholds ($c_{0.4}$ and $e_{0.4}$) degrade the performance again marginally as compared with $1/\infty$ in most cases, implying that further increase of training instances hurts the source dependency model by introducing wrong dependency relations. Overall, e_t works better than c_t , which may be because e_t increases training instances more moderately than c_t does (shown in Table 3) and therefore introduces fewer wrong dependency relations.

6 Discussion and Future Work

Though the experiments are conducted on BTG, the method can be applied to other MT systems which

	MT-03	MT-05
baseline	0.2599	0.2612
$+SDM(1/\infty)$	0.2645	0.2666
$+SDM(c_{0.05})$	0.2626	0.2649
$+SDM(c_{0.1})$	0.2670	0.2666
$+SDM(c_{0.2})$	0.2664	0.2679
$+SDM(c_{0.4})$	0.2646	0.2640
$+SDM(e_{0.05})$	0.2659	0.2640
$+SDM(e_{0.1})$	0.2660	0.2695
$+SDM(e_{0.2})$	0.2652	0.2680
$+SDM(e_{0.4})$	0.2677	0.2655

Table 4: Experimental results on Chinese-to-English translation with the source dependency model (SDM). Statistically significant results (p < 0.01) over the base-line are in **bold**.

use generic synchronous context-free grammars. All we need to do is to build dependency trees in line with synchronous grammars from word alignments following the steps in section 2. If the synchronous grammars are not binary grammars, we do not even need to binarize trees.

There are two promising areas where our work can be extended.

- *Obtain hierarchical alignments using ITG* (Wu, 1997; Zhang and Gildea, 2005). It is clearly shown in the experiments that noises in word alignments influence the performance of SDM considerably. To address this problem, we would explore hierarchical alignments as they contain hierarchical structures intrinsically and would enable us to obtain high quality source dependency trees without using complicated transformation.
- Induce a target dependency model from word alignments. Based on our proposed method, we can also obtain target dependency trees in a similar way as the source dependency trees to produce a target dependency model. This is different from (Shen et al., 2008)'s way of inducing a target dependency language model in that we do not require an external dependency parser.

7 Related Work

In the formally syntax-based machine translation, Chiang (2005) propose to add a "constituent feature" to the log-linear model in order to reward hypotheses under which the hidden hierarchical tree h respects linguistic structures of source language. While no success is seen with this feature, (Marton and Resnik, 2008) and (Chiang et al., 2008) advance this effort and find that penalizing violations of syntactic boundaries of source language improves performance significantly.

In the realm of linguistically syntax-based machine translation, the hidden hierarchical tree becomes more visible because it is explicitly constructed using source-language grammars (Liu et al., 2006; Huang et al., 2006; Zhang et al., 2007) or target-language grammars (Galley et al., 2006; Marcu et al., 2006; Shen et al., 2008). The modeling of h is tightly coupled with the probabilistic source/target-oriented synchronous grammars.

All the prior work listed above requires linguistically motivated parsing. The big distinction from their work is that we build our source dependency model from word alignments without using any monolingual parsing.

8 Conclusion

We have presented a novel method to induce a source dependency model from word alignments without using any linguistic analysis. The induced source dependency model captures dependency relations among source words. We employ this new model in MT to guide the search for better targets along hidden trees with higher probabilities. Our experimental results demonstrate that the integration of the source dependency model into a BTG-based system improves performance significantly on Chineseto-English translation tasks.

Acknowledgements

Special Thanks to Zhang Hao for answering queries on the shift-reduce algorithm.

References

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

- David Chiang. 2005. A Hierarchical Phrase-based Model for Statistical Machine Translation. In *Proceedings of ACL*, pages 263–270.
- David Chiang, Yuval Marton and Philip Resnik. 2008. Online Large-Margin Training of Syntactic and Structural Translation Features. In *Proceedings of EMNLP*.
- Michael Collins. 1999. Head-Driven Statistical Models for Natural Language Parsing. PhD thesis, University of Pennsylvania.
- M. Galley, J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang, and I. Thayer. 2006. Scalable Inference and Training of Context-Rich Syntactic Models. In *Proceedings of ACL-COLING*, 2006.
- Liang Huang, Kevi Knight and Aravind Joshi. 2006. Statistical Syntax-directed Translation with Extended Domain of Locality. In *Proceedings of AMTA 2006*.
- Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of EMNLP*.
- Philipp Koehn, Amittai Axelrod, Alexandra Birch Mayne, Chris Callison-Burch, Miles Osborne and David Talbot. 2005. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. In *International Workshop on Spoken Language Translation*.
- Yang Liu, Qun Liu, Shouxun Lin. 2006. Tree-to-String Alignment Template for Statistical Machine Translation. In Proceedings of ACL-COLING 2006.
- Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phraases. In *Proceedings of EMNLP*.
- Yuval Marton and Philip Resnik. 2008. Soft Syntactic Constraints for Hierarchical Phrase-Based Translation. In *Proceedings of ACL*.
- Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In *Proceedings of ACL 2000*.
- Franz Josef Och and Hermann Ney. 2002. Discriminative Training and Maximum Entropy Models for Statistical Nachine Translation. In *Proceedings of ACL 2002*.
- Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of ACL 2003*.
- Libin Shen, Jinxi Xu and Ralph Weischedel. 2008. A New String-to-Dependency Machine Translation Algorithm with a Target Dependency Language Model. In *Proceedings of ACL-HLT*.
- Andreas Stolcke. 2002. SRILM an extensible language modeling toolkit. In Proceedings of International Conference on Spoken Language Processing, volume 2, pages 901-904.
- Dekai Wu. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377-403.
- Deyi Xiong, Qun Liu and Shouxun Lin. 2006. Maximum Entropy Based Phrase Reordering Model for Statistical Machine Translation. In *Proceedings of ACL-COLING 2006*.
- Nianwen Xue and Fei Xia. 2000. The Bracketing Guidelines for Chinese Treebank Project. Technical Report IRCS 00-08, University of Pennsylvania.
- Hao Zhang, Daniel Gildea, and David Chiang. 2008. Extracting Synchronous Grammars Rules from Word-Level Alignments in Linear Time. In *Proceeding of COLING*.

- Hao Zhang, Daniel Gildea. 2008. Stochastic Lexicalized Inversion Transduction Grammar for Alignment. In *Proceeding* of ACL.
- Min Zhang, Hongfei Jiang, Aiti Aw, Jun Sun, Sheng Li, and Chew Lim Tan. 2007. A Tree-to-Tree Alignment-based Model for Statistical Machine Translation. In *Proceeding* of *MT Summit XI*.