Linguistically Annotated BTG for Statistical Machine Translation

Deyi Xiong, Min Zhang, Aiti Aw and Haizhou Li
Human Language Technology
Institute for Infocomm Research
21 Heng Mui Keng Terrace, Singapore 119613
{dyxiong, mzhang, aaiti}@i2r.a-star.edu.sg

Abstract

Bracketing Transduction Grammar (BTG) is a natural choice for effective integration of desired linguistic knowledge into statistical machine translation (SMT). In this paper, we propose a Linguistically Annotated BTG (LABTG) for SMT. It conveys linguistic knowledge of source-side syntax structures to BTG hierarchical structures through linguistic annotation. From the linguistically annotated data, we learn annotated BTG rules and train linguistically motivated phrase translation model and reordering model. We also present an annotation algorithm that captures syntactic information for BTG nodes. The experiments show that the LABTG approach significantly outperforms a baseline BTG-based system and a state-of-the-art phrase-based system on the NIST MT-05 Chinese-to-English translation task. Moreover, we empirically demonstrate that the proposed method achieves better translation selection and phrase reordering.

1 Introduction

Formal grammar used in statistical machine translation (SMT), such as Bracketing Transduction Grammar (BTG) proposed by (Wu, 1997) and the synchronous CFG presented by (Chiang, 2005), provides a natural platform for integrating linguistic knowledge into SMT because hierarchical structures produced by the formal grammar resemble linguistic structures.[1] Chiang (2005) attempts to integrate linguistic information into his formally syntax-based system by adding a constituent feature. Unfortunately, the linguistic feature does not show significant improvement on the test set. In this paper, we further this effort by integrating linguistic knowledge into BTG.

We want to augment BTG's formal structures with linguistic structures since they are both hierarchical. In particular, our goal is to learn a more linguistically meaningful BTG from real-world bitexts by projecting linguistic structures onto BTG formal structures. In doing so, we hope to (1) maintain the strength of phrase-based approach since phrases are still used on BTG leaf nodes; (2) obtain a tight integration of linguistic knowledge in the translation model; (3) and finally avoid inducing a complicated linguistic synchronous grammar with expensive computation. The challenge, of course, is that BTG hierarchical structures are not always aligned with the linguistic structures in the syntactic parse trees of source or target language.

Along this line, we propose a novel approach: Linguistically Annotated BTG (LABTG) for SMT. The LABTG annotates BTG rules with *linguistic elements* that are learned from syntactic parse trees on the source side through an annotation algorithm, which is capable of labelling both syntactic and non-syntactic phrases. The linguistic elements extracted from parse trees capture both internal lexical content and external context of phrases. With these linguistic annotations, we expect the LABTG to address two traditional issues of standard phrase-based SMT (Koehn et al., 2003) in a more effective manner. They are (1) phrase translation: translating phrases according to their contexts; (2) phrase reordering: incorporating richer linguistic features for better reordering.

The proposed LABTG displays two unique characteristics when compared with BTG-based SMT (Wu, 1996; Xiong et al., 2006). The first is that two linguistically-informed sub-models are introduced for better phrase translation and reordering: annotated phrase translation model and

[1] We inherit the definitions of *formal* and *linguistic* from (Chiang, 2005) which makes a distinction between formally syntax-based SMT and linguistically syntax-based SMT.

Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 1009–1016
Manchester, August 2008

annotated reordering model. The second is that our proposed annotation algorithm and scheme are capable of conveying linguistic knowledge from source-side syntax structures to BTG structures. We describe the LABTG model and the annotation algorithm in Section 4. To better explain the LABTG model, we establish a unified framework of BTG-based SMT in Section 3. We conduct a series of experiments to study the effect of the LABTG in Section 5.

2 Related Work

There have been various efforts to integrate linguistic knowledge into SMT systems, either from the target side (Marcu et al., 2006; Hassan et al., 2007; Zollmann and Venugopal, 2006), the source side (Quirk et al., 2005; Liu et al., 2006; Huang et al., 2006) or both sides (Eisner, 2003; Ding et al., 2005; Koehn and Hoang, 2007), just to name a few. LABTG can be considered as, but not limited to, a new attempt that enriches translation model with source-side linguistic annotations.

(Huang and Knight, 2006) and (Hassan et al., 2007) introduce relabeling and supertagging on the target side, respectively. The former re-annotates syntactified phrases to learn grammatical distinctions while the latter supertags standard plain phrases, both applied on the target side. The difference between their work and LABTG is significant because we annotate standard plain phrases using linguistic elements on the source side. Compared with the target side annotation which improves fluency and grammaticality of translation output, linguistic annotation on the source side helps to improve translation adequacy.

Recently, some researchers have extended and created several variations of BTG/ITG. Zhang et al. (2005) propose lexicalized ITG for better word alignment. Xiong et al. (2006) demonstrate that their MEBTG, a BTG variation with MaxEnt-based reordering model, can improve phrase reordering significantly. Similarly, Setiawan et al. (2007) use an enhanced BTG variation with function words for reordering. LABTG differs from these BTG variations in that the latter does not use any external linguistic knowledge.

Zhang et al. (2007) describe a phrase reordering model based on BTG-style rules which integrates source-side syntactic knowledge. Our annotated reordering model of LABTG differs from their work in two key aspects. Firstly, we allow any phrase reorderings while they only reorder syntactic phrases. In their model, only syntactic phrases can use linguistic knowledge from parse trees for reordering while non-syntactic phrases are combined monotonously with a constant reordering score since no syntactic knowledge can be used at all. Our proposed LABTG successfully overcomes this limitation by supporting linguistic annotation on both syntactic and non-syntactic phrases. Moreover, we show that excluding non-syntactic phrase from reordering does hurt the performance. Secondly, we use richer linguistic knowledge in reordering, including head words and syntactic labels of context nodes, when compared with their model. Our experiments show that these additional information can improve reordering.

3 BTG Based SMT

We establish a unified framework for BTG-based SMT in this section. There are two kinds of rules in BTG, lexical rules (denoted as r^l) and merging rules (denoted as r^m):

$$r^l : A \to x/y$$

and

$$r^m : A \to [A_l, A_r] | \langle A_l, A_r \rangle$$

Lexical rules translate source phrase x into target phrase y and generate a leaf node A in BTG tree. Merging rules combine left and right neighboring phrases A_l and A_r into a larger phrase A in an order $o \in \{straight, inverted\}$.

We define a BTG derivation D as a sequence of independent applications of lexical and merging rules ($D = \langle r^l_{1..n_l}, r^m_{1..n_m} \rangle$). Given a source sentence, the decoding task of BTG-based SMT is to find a best derivation, which yields the best translation.

Similar to (Xiong et al., 2006), we can assign a probability to each rule using a log-linear model with different features and corresponding λ weights, then multiply them to obtain $P(D)$. For convenience of notation and keeping in line with the common understanding of standard phrase-based model, here we re-organize these features into translation model (P_T), reordering model (P_R) and target language model (P_L) as follows

$$P(D) = P_T(r^l_{1..n_l}) \cdot P_R(r^m_{1..n_m})^{\lambda_R} \cdot P_L(e)^{\lambda_L} \cdot exp(|e|)^{\lambda_w} \quad (1)$$

where $exp(|e|)$ is the word penalty.

The translation model is defined as:

$$P_T(r^l_{1..n_l}) = \prod_{i=1}^{n_l} P(r^l_i)$$

$$P(r^l) = p(x|y)^{\lambda_1} \cdot p(y|x)^{\lambda_2} \cdot p_{lex}(x|y)^{\lambda_3} \cdot p_{lex}(y|x)^{\lambda_4} \cdot exp(1)^{\lambda_5} \quad (2)$$

where $p(\cdot)$ represent the phrase translation probabilities in both directions, $p_{lex}(\cdot)$ denote the lexical translation probabilities in both directions, and $exp(1)$ is the phrase penalty.

Similarly, the reordering model is defined on the merging rules as follows

$$P_R(r^m_{1..n_m}) = \prod_{i=1}^{n_m} P(r^m_i) \quad (3)$$

In the original BTG model (Wu, 1996), $P(r^m)$ was actually a prior probability which can be set based on the order preference of the language pairs. In MEBTG (Xiong et al., 2006), however, the probability is calculated more sophisticatedly using a MaxEnt-based classification model with boundary words as its features.

4 Linguistically Annotated BTG Based SMT

We extend the original BTG into the linguistically annotated BTG by adding linguistic annotations from source-side parse trees to both BTG lexical rules and merging rules. Before we elaborate how the LABTG extends the baseline, we introduce annotated BTG rules.

In the LABTG, both lexical rules and merging rules are annotated with linguistic elements as follows

$$ar^l : A^a \rightarrow x\#a/y$$

and

$$ar^m : A^a \rightarrow [A^{a_l}_l, A^{a_r}_r] | \langle A^{a_l}_l, A^{a_r}_r \rangle$$

The annotation a comprises three linguistic elements from source-side syntactic parse tree: (1) head word hw, (2) the part-of-speech (POS) tag ht of head word and (3) syntactic label sl. In annotated lexical rules, the three elements are combined together and then attached to x as an annotation unit. In annotated merging rules, each node involved in merging is annotated with these three elements individually.

There are various ways to learn the annotated rules from training data. The straight-forward way is to first generate the best BTG tree for each sentence pair using the way of (Wu, 1997), then annotate each BTG node with linguistic elements by projecting source-side syntax tree to BTG tree, and finally extract rules from these annotated BTG trees. This way restricts learning space to only the best BTG trees[2], and leads to the loss of many useful annotated rules.

Therefore, we use an alternative way to extract the annotated rules as illustrated below. Firstly, we run GIZA++ (Och and Ney, 2000) on the training corpus in both directions and then apply the “grow-diag-final” refinement rule (Koehn et al., 2003) to obtain many-to-many word alignments. Secondly, we extract bilingual phrases from the word-aligned corpus, then annotate their source sides with linguistic elements to obtain the annotated lexical rules.[3] Finally, we learn reordering examples (Xiong et al., 2006), annotate their two neighboring sub-phrases and whole phrases, and then generalize them in the annotated merging rules. Although this alternative way may also miss reorderings due to word alignment errors, it is still more flexible and robust than the straight-forward one, and can learn more annotated BTG rules without constructing BTG trees explicitly.

4.1 LABTG Annotation Algorithm

During the process of rule learning and decoding, we need to annotate bilingual phrases or BTG nodes generated by the decoder given a source sentence together with its parse tree. Since both phrases and BTG nodes can be projected to a span on the source sentence, we run our annotation algorithm on source-side spans and then assign annotation results to the corresponding phrases or nodes. If the span is exactly covered by a single subtree in the source-side parse tree, it is called **syntactic span**, otherwise **non-syntactic span**. One of the challenges in this annotation algorithm is that BTG nodes (or phrases) are not always covering syntactic span, in other words, are not always aligned to constituent nodes in the source-side tree. To solve this problem, we use heuristic rules to generate pseudo head word and **composite label** which consists of syntactic labels of three relevant constituents for the non-syntactic span.

The annotation algorithm is shown in Fig. 1. For a syntactic span, the annotation is trivial. Annotation elements directly come from the subtree that exactly covers the span. For a non-syntactic

[2] Producing BTG forest for each sentence pair is very time-consuming.

[3] This makes the number of extracted annotated lexical rules proportional to that of bilingual phrases.

```
1:  Annotator (span $s = \langle i, j \rangle$, source-side parse tree $t$)
2:  **if** $s$ is a syntactic span **then**
3:      Find the subtree $c$ in $t$ which exactly covers $s$
4:      $s.a := \{c.hw, c.ht, c.sl\}$
5:  **else**
6:      Find the smallest subtree $c^*$ subsuming $s$ in $t$
7:      **if** $c^*.hw \in s$ **then**
8:          $s.a.hw := c^*.hw$ and $s.a.ht := c^*.ht$
9:      **else**
10:         Find the word $w \in s$ which is nearest to $c^*.hw$
11:         $s.a.hw := w$ and $s.a.ht := w.t$  /*$w.t$ is the POS
            tag of $w$*/
12:     **end if**
13:     Find the left context node $ln$ of $s$ in $c^*$
14:     Find the right context node $rn$ of $s$ in $c^*$
15:     $s.a.sl := ln.sl$-$c^*.sl$-$rn.sl$
16: **end if**
```

Figure 1: The LABTG Annotation Algorithm.

span, the process is much complicated. Firstly, we need to locate the smallest subtree c^* subsuming the span (line 6). Secondly, we try to identify the head word/tag of the span (line 7-12) by using c^*'s head word hw directly if it is within the span. Otherwise, the word within the span which is nearest to hw will be assigned as the head word of the span. Finally, we determine the composite label of the span (line 13-15), which is formulated as L-C-R. L/R refers to the syntactic label of the left/right **context node** of s which is a sub-node of c^*. There are different ways to define the context node of a span in the source-side parse tree. It can be the closest neighboring node or the boundary node which is the highest leftmost/rightmost sub-node of c^* not overlapping the span. If there is no such context node (the span s is exactly aligned to the left/right boundary of c^*), L/R will be set to NULL. C is the label of c^*. L, R and C together define the external syntactic context of s.

Fig. 2 shows a syntactic parse tree for a Chinese sentence, with head word annotated for each internal node.[4] Some sample annotations are given in Table 1. We also show different composite labels for non-syntactic spans with different definitions of their context nodes. sl_1 is obtained when the boundary node is defined as the context node while sl_2 is obtained when the closest neighboring node is defined as the context node.

4.2 LABTG Model

To better model annotated rules, the LABTG contributes two significant modifications to formula (1). First is the annotated phrase translation model with source side linguistically enhanced to replace the standard phrase translation model, and second is the additional MaxEnt-based reordering model that uses linguistic annotations as features. The LABTG model is formulated as follows

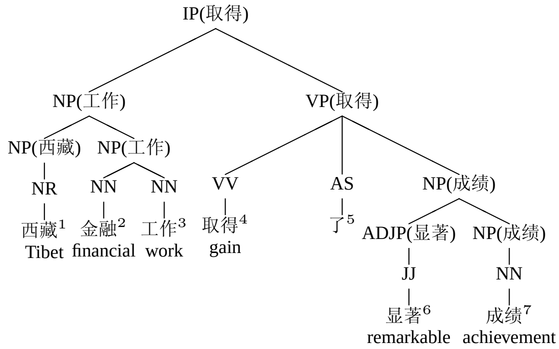

Figure 2: A syntactic parse tree with head word annotated for each internal node. The superscripts of leaf nodes denote their surface positions from left to right.

$span$	hw	ht	sl_1 (boundary node)	sl_2 (neighboring node)
⟨1, 2⟩	金融	NN	NULL-NP-NN	NULL-NP-NN
⟨2, 3⟩	工作	NN	NP	NP
⟨2, 4⟩	取得	VV	NP-IP-NP	NP-IP-AS
⟨3, 4⟩	取得	VV	NP-IP-NP	NN-IP-AS

Table 1: Annotation samples according to the tree shown in Fig. 2. hw/ht represents head word/tag, respectively. sl means the syntactic label.

$$
\begin{aligned}
P(D) &= P_{T_a}(ar^l_{1..n_l}) \cdot P_{R_b}(r^m_{1..n_m})^{\lambda_{R_b}} \\
&\cdot P_{R_a}(ar^m_{1..n_m})^{\lambda_{R_a}} \cdot P_L(e)^{\lambda_L} \cdot exp(|e|)^{\lambda_w} \qquad (4)
\end{aligned}
$$

Here P_{T_a} is the annotated phrase translation model, P_{R_b} is the reordering model from MEBTG using boundary words as features and P_{R_a} is the annotated reordering model using linguistic annotations of nodes as features.

Annotated Phrase Translation Model The annotated phrase translation model P_{T_a} is similar to formula (2) except that phrase translation probabilities on both directions are $p(x\#a|y)$ and $p(y|x\#a)$ respectively, instead of $p(x|y)$ and $p(y|x)$. By introducing annotations into the translation model, we integrate linguistic knowledge into the statistical selection of target equivalents.

Annotated Reordering Model The annotated reordering model P_{R_a} is a MaxEnt-based classification model which uses linguistic elements of each annotated node as its features. The model can be formulated as

$$
P_{R_a}(ar^m) \quad = \quad p_\theta(o|A^a, A^{a_l}_l, A^{a_r}_r)
$$

[4] In this paper, we use phrase labels from the Penn Chinese Treebank (Xue et al., 2005).

$$= \frac{exp(\sum_i \theta_i h_i(o, A^a, A_l^{a_l}, A_r^{a_r}))}{\sum_o exp(\sum_i \theta_i h_i(o, A^a, A_l^{a_l}, A_r^{a_r}))}$$

where the functions $h_i \in \{0, 1\}$ are reordering features and θ_i are weights of these features.

Each merging rule involves 3 nodes $(A^a, A_l^{a_l}, A_r^{a_r})$ and each node has 3 linguistic elements (hw, ht, sl). Therefore, the model has 9 features in total. Taking the left node $A_l^{a_l}$ as an example, the model could use its head word w as feature as follows

$$h_i(o, A^a, A_l^{a_l}, A_r^{a_r}) = \begin{cases} 1, & A_l^{a_l}.hw = w, o = straight \\ 0, & otherwise \end{cases}$$

4.3 Training

To train the annotated translation model, firstly we extract all annotated lexical rules from source-side parsed, word-aligned training data. Then we estimate the annotated phrase translation probabilities $p(x\#a|y)$ and $p(y|x\#a)$ using relative counts from all collected annotated lexical rules. For example, $p(y|x\#a)$ can be calculated as follows

$$p(y|x\#a) = \frac{count(x\#a, y)}{\sum_y count(x\#a, y)}$$

One might think that linguistic annotations would cause serious data sparseness problem and the probabilities should be smoothed. However, according to our statistics (described in the next section), the differences in annotations for the same source phrase x are not so diverse. So we take a direct backoff strategy to map unseen annotated lexical rules to their un-annotated versions on the fly during decoding, which is detailed in the next subsection.

To train the annotated reordering model, we generate all annotated reordering examples, then obtain features using linguistic elements of these examples, and finally estimate feature weights based on the maximum entropy principle.

4.4 Decoding

A CKY-style decoder with beam search is developed, similar to (Xiong et al., 2006). Each input source sentence is firstly parsed to obtain its syntactic tree. Then the CKY-style decoder tries to generate the best annotated BTG tree using the trained annotated lexical and merging rules. We store all annotated lexical rules and their probabilities in a standard phrase table Ω, where source phrases are augmented with annotations. During the application of annotated lexical rules, we label each source phrase x with linguistic annotation a through the annotation algorithm given the source-side parse tree, and retrieve $x\#a$ from Ω. In the case of unseen combination $x\#a$, we map it to x and lookup x in the phrase table so that we can use the un-annotated lexical rule $A \rightarrow x/y$. We set $p(y|x) = max_{a'} p(y|x\#a')$ and $p(x|y) = max_{a'} p(x\#a'|y)$ where $(x, a', y) \in \Omega$. When two neighboring nodes are merged in a specific order, the two reordering models, P_{R_b} and P_{R_a}, will evaluate this merging independently with individual scores. The former uses boundary words as features while the latter uses the linguistic elements as features, annotated on the BTG nodes through the annotation algorithm according to the source-side parse tree.

5 Experiments and Analysis

In this section we conducted a number of experiments to demonstrate the competitiveness of the proposed LABTG based SMT when compared with two baseline systems: Moses (Koehn et al., 2007), a state-of-the-art phrase-based system and MEBTG (Xiong et al., 2006), a BTG based system. We also investigated the impact of different annotation schemes on the LABTG model and studied the effect of annotated phrase translation model and annotated reordering model on translation selection and phrase reordering respectively. All experiments were carried out on the Chinese-to-English translation task of the NIST MT-05 with case-sensitive BLEU scores reported.

The systems were trained on the FBIS corpus. We removed 15,250 sentences, for which the Chinese parser (Xiong et al., 2005) failed to produce syntactic parse trees. The parser was trained on the Penn Chinese Treebank with a F1 score of 79.4%. From the remaining FBIS corpus (224, 165 sentence pairs), we obtained 4.55M standard bilingual phrases (including 2.75M source phrases) for the baseline systems and 4.65M annotated lexical rules (including 3.13M annotated source phrases augmented with linguistic annotations) for the LABTG system using the algorithm mentioned above. These statistics reveal that there are 1.14 (3.13M/2.75M) annotations per source phrase, which means our annotation algorithm does not increase the number of extracted rules exponentially.

We extracted 2.8M reordering examples, from

System	BLEU
Moses	0.2386
MEBTG	0.2498
LABTG	0.2667

Table 2: LABTG vs. Moses and MEBTG.

which we generated 114.8K reordering features for the reordering model P_{R_b} (shared by both MEBTG and LABTG systems) using the right boundary words of phrases and 85K features for the annotated reordering model P_{R_a} (only included in the LABTG system) using linguistic annotations. We ran the MaxEnt toolkit (Zhang, 2004) to tune reordering feature weights with iteration number being set to 100 and Gaussian prior to 1 to avoid overfitting.

We built our four-gram language model using Xinhua section of the English Gigaword corpus (181.1M words) with the SRILM toolkit (Stolcke, 2002). For the efficiency of minimum-error-rate training (Och, 2003), we built our development set (580 sentences) using sentences not exceeding 50 characters from the NIST MT-02 evaluation test data.

5.1 LABTG vs. phrase-based SMT and BTG-based SMT

We compared the LABTG system with two baseline systems. The results are given in Table 2. The LABTG outperforms Moses and MEBTG by 2.81 and 1.69 absolute BLEU points, respectively. These significant improvements indicate that BTG formal structures can be successfully extended with linguistic knowledge extracted from syntactic structures without losing the strength of phrase-based method.

5.2 The Effect of Different Annotation Schemes

A great amount of linguistic knowledge is conveyed through the syntactic label sl. To obtain this label, we tag syntactic BTG node with single label C from its corresponding constituent in the source-side parse tree while annotate non-syntactic BTG node with composite label formulated as L-C-R. We conducted experiments to study the effect of different annotation schemes on the LABTG model by comparing three different annotation schemes for non-syntactic BTG node: (1) using single label C from its corresponding smallest subtree c^* (C), (2) constructing composite label using neighboring node as context node (N-C-N), and (3) constructing composite label using boundary node as context node (B-C-B). The results are shown in Table 3.

Annotation scheme	BLEU
C	0.2626
N-C-N	0.2591
B-C-B	0.2667
Annotating syntactic nodes with composite label	0.2464

Table 3: Comparison of different annotation schemes.

On the one hand, linguistic annotation provides additional information for LABTG, transferring knowledge from source-side linguistic structures to BTG formal structures. On the other hand, however, it is also a constraint on LABTG, guiding the annotated translation model and reordering model to the selection of target alternatives and reordering patterns, respectively. A tight constraint always means that annotations are too specific, although they incorporate rich knowledge. Too specific annotations are more sensitive to parse errors, and easier to make the model lose correct translations or use wrong reordering patterns. That is the reason why the annotation scheme “N-C-N” and “Annotating syntactic nodes with composite label” [5] both hurt the performance. Conversely, a loose constraint means that annotations are too generic and have less knowledge incorporated. The annotation scheme “C” is such a scheme with loose constraint and less knowledge.

Therefore, an ideal annotation scheme should not be too specific or too generic. The annotation scheme “B-C-B” achieves a reasonable balance between knowledge incorporation and constraint, which obtains the best performance. Therefore we choose boundary node as context node for label annotation of non-syntactic BTG nodes in experiments described later.

5.3 The Effect of Annotated Translation Model

To investigate the effect of the annotated translation model on translation selection, we compared the standard phrase translation model P_T used in MEBTG with the annotated phrase translation

[5] In this annotation scheme, we produce composite label L-C-R for both syntactic and non-syntactic BTG nodes. For syntactic node, sibling node is used as context node while for non-syntactic node, boundary node is used as context node.

Translation model	BLEU
P_T	0.2498
P_{T_a}	0.2581
P_{T_a} (-NULL)	0.2548

Table 4: The effect of annotated translation model.

model P_{T_a}. The experiment results are shown in Table 4. The significant improvement in the BLEU score indicates that the annotated translation model helps to select better translation options.

Our study on translation output shows that annotating phrases with source-side linguistic elements can provide at least two kinds of information for translation model to improve the adequacy: category and context. The category knowledge of a phrase can be used to select its appropriate translation related to its category. For example, Chinese phrase “值” can be translated into “value” if it is a verb or “at/on” if it is a proposition. However, the baseline BTG-based system always selects the proposition translation even if it is a verb because the language model probability for proposition translation is higher than that of verb translation. This wrong translation of content words is similar to the incorrect omission reported in (Och et al., 2003), which both hurt translation adequacy. The annotated translation model can avoid wrong translation by filtering out phrase candidates with unmatched categories.

The context information (provided by context node) is also quite useful for translation selection. Even the “NULL” context, which we used in label annotation to indicate that a phrase is located at the boundary of a constituent, provides some information, such as, transitive or intransitive attribute of a verb phrase. The last row of Tabel 4 shows that if we remove “NULL” in label annotation, the performance is degraded. (Huang and Knight, 2006) also reported similar result by using sisterhood annotation on the target side.

5.4 The Effect of Annotated Reordering Model

To investigate the effect of the annotated reordering model, we integrate P_{R_a} with various settings in MEBTG while keeping its original phrase translation model P_T and reordering model P_{R_b} unchanged. We augment P_{R_a}'s feature pool incrementally: firstly using only single labels [6](SL) as features (132 features in total), then constructing composite labels for non-syntactic phrases (+BNL) (6.7K features), and finally introducing head words into the feature pool (+BNL+HWT) (85K features). This series of experiments demonstrate the impact and degree of contribution made by each feature for reordering. We also conducted experiments to investigate the effect of restricting reordering to syntactic phrases using the best reordering feature set (SL+BNL+HWT) for P_{R_a}. The experimental results are presented in Table 2, from which we have the following observations:

Reordering Configuration	BLEU
P_{R_b}	0.2498
P_{R_b} + P_{R_a} (SL)	0.2588
P_{R_b} + P_{R_a} (+BNL)	0.2627
P_{R_b} + P_{R_a} (+BNL+HWT)	0.2652
P_{R_b} + P_{R_a} (SL+BNL+HWT): only allowed syntactic phrase reordering	0.2512

Table 5: The effect of annotated reordering model.

(1) Source-side syntactic labels (SL) capture reordering patterns between source structures and their target counterparts. Even when the baseline feature set SL with only 132 features is used for P_{R_a}, the BLEU score improves from 0.2498 to 0.2588. This is because most of the frequent reordering patterns between Chinese and English have been captured using syntactic labels. For example, the pre-verbal modifier PP in Chinese is translated into post-verbal counterpart in English. This reordering can be described by a rule with an inverted order: $VP \rightarrow \langle PP, VP \rangle$, and captured by our syntactic reordering features.

(2) Context information, provided by labels of context nodes (BNL) and head word/tag pairs (HWT), also improves phrase reordering. Producing composite labels for non-syntactic BTG nodes (+BNL) and integrating head word/tag pairs into P_{R_a} as reordering features (+BNL+HWT) are both effective, indicating that context information complements syntactic label for capturing reordering patterns.

(3) Restricting phrase reordering to syntactic phrases is harmful. The BLEU score plummets from 0.2652 to 0.2512.

6 Conclusions

In this paper, we have presented a Linguistically Annotated BTG based approach to effectively integrate linguistic knowledge into SMT by merging

[6] For non-syntactic node, we only use the single label C, without constructing composite label L-C-R.

source-side linguistic structures with BTG hierarchical structures. The LABTG brings BTG-based SMT towards linguistically syntax-based SMT and narrows the linguistic gap between them. Our experimental results show that the LABTG significantly outperforms the state-of-the-art phrase-based SMT and the baseline BTG-based SMT. The proposed method also offers better translation selection and phrase reordering by introducing the annotated phrase translation model and the annotated reordering model with linguistic annotations.

We conclude that (1) source-side syntactic information can improve translation adequacy; (2) linguistic annotations of BTG nodes well capture reordering patterns between source structures and their target counterparts; (3) integration of linguistic knowledge into SMT should be carefully conducted so that the incorporated knowledge could not have negative constraints on the model[7].

References

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

Yuan Ding and Martha Palmer. 2005. Machine Translation Using Probabilistic Synchronous Dependency Insertion Grammars. In *Proceedings of ACL 2005*.

Jason Eisner. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of ACL 2003*.

Hany Hassan, Khalil Sima'an and Andy Way. 2007. Supertagged Phrase-based Statistical Machine Translation. In *Proceedings of ACL 2007*.

Bryant Huang, Kevi Knight. 2006. Relabeling Syntax Trees to Improve Syntax-Based Machine Translation Quality. In *Proceedings of NAACL-HLT 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, Franz Joseph Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proceedings of HLT/NAACL*.

Philipp Koehn, Hieu Hoang. 2007. Factored Translation Models. In *Proceedings of EMNLP 2007*.

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. ACL 2007, demonstration session, Prague, Czech Republic, June 2007.

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*.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of ACL 2000*.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of ACL 2003*.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin, Dragomir Radev. 2003. Final Report of Johns Hopkins 2003 Summer Workshop on Syntax for Statistical Machine Translation.

Chris Quirk, Arul Menezes and Colin Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. In *Proceedings of ACL 2005*.

Hendra Setiawan, Min-Yen Kan and Haizhou Li. 2007. Ordering Phrases with Function Words. In *Proceedings of ACL 2007*.

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. 1996. A Polynomial-Time Algorithm for Statistical Machine Translation. In *Proceedings of ACL 1996*.

Dekai Wu. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3):377-403.

Deyi Xiong, Shuanglong Li, Qun Liu, Shouxun Lin, Yueliang Qian. 2005. Parsing the Penn Chinese Treebank with Semantic Knowledge. In *Proceedings of IJCNLP*, Jeju Island, Korea.

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, 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):207-238.

Dongdong Zhang, Mu Li, Chi-Ho Li and Ming Zhou. 2007. Phrase Reordering Model Integrating Syntactic Knowledge for SMT. In *Proceedings of EMNLP-CoNLL 2007*.

Hao Zhang and Daniel Gildea. 2005. Stochastic Lexicalized Inversion Transduction Grammar for Alignment. In *Proceedings of ACL 2005*.

Le Zhang. 2004. Maximum Entropy Modeling Tooklkit for Python and C++. Available at http://homepages.inf.ed.ac.uk/s0450736 /maxent_toolkit.html.

Andreas Zollmann and Ashish Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. In NAACL 2006 - Workshop on statistical machine translation, New York. June 4-9.

[7] For example, the annotation scheme "N-C-N" incorporates rich syntactic knowledge, but also tightens the constraint on the model, which therefore loses robustness.