

# Can Semantic Roles Improve Syntax-Based Machine Translation?

Ding Liu and Daniel Gildea  
Department of Computer Science  
University of Rochester  
Rochester, NY 14627

## Abstract

This paper compares the performance of a Tree-to-string (TTS) transducer with automatically generated/gold-standard parse trees and semantic roles. Experimental results show that improving the parsing quality can lead to significant improvement in MT performance and adding semantic roles in the syntax tree labels does not improve the TTS transducer. Another approach of using semantic roles: *skeleton* template extraction, is proposed and shown to be better than extracting straight long templates down to a certain depth.

## 1 Introduction

Syntax-based statistical machine translation (SSMT) has achieved significant progress during recent years, with two threads developing simultaneously: the synchronous parsing-based SSMT (Galley et al., 2006; May and Knight, 2007) and the tree-to-string (TTS) transducer (Liu et al., 2006; Huang et al., 2006). Synchronous SSMT here denotes the systems which has a plain source sentence as the input and generates the translation and the syntactic structure for both the source and the translation simultaneously. Such systems are sometimes also called TTS transducers, but in this paper, TTS transducer refers to the system which starts with the syntax tree of a source sentence and recursively transforms the tree to the target language based on TTS templates.

In synchronous SSMT, TTS templates are used similar to the context free grammar used in the standard CYK parser, thus the syntax is part of the output and can be thought of as a constraint on the translation process. In the TTS transducer, since the parse

tree is given, syntax can be thought of as an additional feature of the input to be used in the translation. The idea of synchronous SSMT can be traced back to Wu (1997)'s Stochastic Inversion Transduction Grammars. A systematic method for extracting TTS templates from parallel corpora was proposed by Galley et al. (2004), and later binarized by Zhang et al. (2006) for high efficiency and accuracy. In the other track, the TTS transducer originated from the tree transducer proposed by Rounds (1970) and Thatcher (1970) independently. Graehl and Knight (2004) generalized the tree transducer to the TTS transducer and introduced an EM algorithm to estimate the probability of TTS templates based on a bilingual corpus with one side parsed. Liu et al. (2006) and Huang et al. (2006) then used the TTS transducer on the task of Chinese-to-English and English-to-Chinese translation, respectively, and achieved decent performance. ?) further improved the TTS transducer by using different template normalization and incorporating an  $n$ -gram-based tree decomposition model.

Inspired by the progress made in SSMT, its natural extension, semantics-based SSMT is an interesting approach to explore. ?) attempted this by replacing the syntax tree labels in a TTS transducer with the shallow semantics-based labels of PropBank (Palmer et al., 2005). Using semantic roles in a TTS transducer is quite straight-forward: first, semantic role labeling (SRL) is done for the syntax trees; then, the syntax tree labels are replaced with the semantic role-based labels; at last, the semantics-based trees are used in both template extraction and decoding. Nothing except the input data needs be changed for a TTS transducer to adapt itself to a semantics-based TTS transducer. With a

synchronous SSMT, it is not obvious how to integrate the semantic roles into the system except the re-ranking of the top-k outputs based on semantic role features, because the reliable semantic role labeling consumes the sentence-wide features which cannot be obtained before the whole syntax tree is present (Xue and Palmer, 2004; Ward et al., 2004; Toutanova et al., 2005). ?’s approach showed some but not always positive effects of the semantic roles based on a test set of 500 sentences, leading us wonder about the significance of the improvement brought by semantic roles.

This paper implements a more systematic version of the semantics-based TTS transducer, where the semantics-based template probabilities are smoothed with and backed-off to the syntax-based template probabilities if necessary. Also  $N$ -fold cross validation and re-sampling based statistical significance test (Koehn, 2004) is used to examine the effect of semantic roles on MT performance. Unfortunately, semantic roles does not bring any improvement over the syntax-based TTS transducer. To find out whether this is due to the low accuracy of the automatic SRL, a parallel corpus with gold-standard parses and semantic roles is used to conduct the following comparison experiments:

1. Automatic parse vs. gold-standard parse in terms of MT performance.
2. semantics-based TTS transducer vs. syntax-based TTS transducer with gold-standard parse and semantic roles.

The results indicate that parsing accuracy has a strong impact on the MT performance: the better the parses are, the better performance of the TTS transducer; and the semantic roles do not improve much even with the gold-standard parses and semantic roles. By comparing the distribution of the right hand-sides (RHS) given the same left hand-side (LHS) with or without the semantic roles, we find out that only a small portion of the semantics-based templates have very different distribution from their syntactic counterparts, which is why the semantics-based TTS transducer does not get very different outputs from the syntax-based ones.

To make full use of the correspondence of the verbs and their roles, an approach is proposed to ex-

tract skeleton templates based on the semantic roles. Skeleton templates contain a verb and all its semantic roles, and are meant to address the major re-ordering of a sentence. Skeleton templates are shown to be better than template going all the way down to depth 6 with gold-standard parses/semantic roles, but they are not very robust against automatic parses/semantic roles.

The remainder of the paper is organized as follows: Section 2 describes the TTS transducer we experiment with; Section 3 describes the two approaches utilizing semantic roles in a TTS transducer; Section 4 presents the experimental results with automatic/gold-standard parses and semantic roles; Section 5 discusses some other ways of using semantic roles, and Section 6 gives the conclusion.

## 2 A Basic Tree-to-string Transducer for Machine Translation

A TTS transducer receives a syntax tree as its input and, by recursively using TTS templates, generates the target string. A TTS template is composed of a left-hand side (LHS) and a right-hand side (RHS), where LHS is a subtree pattern and RHS is a sequence of the variables and translated words. The variables in the RHS of a template correspond to the bottom level non-terminals in the LHS’s subtree pattern, and their relative order indicates the permutation desired at the point where the template is applied to translate one language to another. The variables are further transformed, and the recursive process goes on until there are no variables left. The formal description of a TTS transducer is given by Graehl and Knight (2004), and our baseline approach follows the *Extended Tree-to-String Transducer* defined by Huang et al. (2006). Figure 1 gives an example of the English-to-Chinese TTS template, which shows how to translate a skeleton YES/NO question from English to Chinese.  $NP^1$  and  $VP^2$  are the variables whose relative positions in the translation are determined by the template while their actual translations are still unknown and dependent on the subtrees rooted at them; and the English words *Is* and *not* are translated into the Chinese word *MeiYou* in the context of the template. The superscripts attached to the variables are used to distinguish the non-terminals with identical names

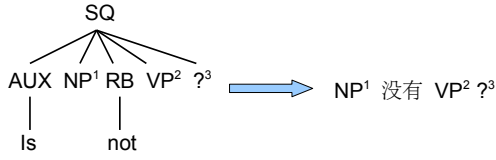


Figure 1: A TTS Template Example

(if there are any). For a given derivation (decomposition) of a syntax tree, the translation probability is computed as the product of the templates which generate both the source syntax trees and the target translations. In theory, the translation model should sum over all possible derivations generating the target translation, but in practice, usually only the best derivation is considered:

$$Pr(S | T, D^*) = \prod_{t \in D^*} Weight(t)$$

Here,  $S$  denotes the target translation,  $T$  denotes the source syntax tree, and  $D^*$  denotes the best derivation of  $T$ . The implementation of a TTS transducer can be done either top down with memoization to the visited subtrees (Huang et al., 2006), or with a bottom-up dynamic programming (DP) algorithm (Liu et al., 2006). This paper uses the latter approach, and the algorithm is sketched in Figure 2. For the baseline approach, only the translation model and  $n$ -gram model for the target language are used:

$$S^* = \operatorname{argmax}_S Pr(T|S) = \operatorname{argmax}_S Pr(S)Pr(S|T)$$

Since the  $n$ -gram model tends to favor short translations, a penalty is added to the translation templates with fewer RHS symbols than LHS leaf symbols:

$$Penalty(t) = \exp(|t.RHS| - |t.LHSLeaf|)$$

where  $|t.RHS|$  denotes the number of symbols in the RHS of  $t$ , and  $|t.LHSLeaf|$  denotes the number of leaves in the LHS of  $t$ . The length penalty is analogous to the length feature widely used in log-linear models for MT (Huang et al., 2006; Liu et al., 2006; Och and Ney, 2004). Here we distribute the penalty into TTS templates for the convenience of DP, so that we don't have to generate the  $N$ -best list

and do re-ranking. To speed up the decoding, standard beam search is used.

In Figure 2, *BinaryCombine* denotes target-size binarization (Huang et al., 2006). The translation candidates of the template's variables, as well as its terminals, are combined pair-wise in the order they appear in the RHS of the template.  $f_i$  denotes a combined translation, whose probability is equal to the product of the probabilities of the component translations, the probability of the rule, the  $n$ -gram probability of connecting the component translations, and the length penalty of the template.  $\alpha$ ,  $\beta$  and  $\gamma$  are the weights of the length penalty, the translation model, and the  $n$ -gram language model, respectively. Each state in the DP chart denotes the best translation of a tree node with a certain *prefix* and *suffix*. The length of the *prefix* and the *suffix* is equal to the length of the  $n$ -gram model minus one. Without beam pruning, the decoding algorithm runs in  $O(N^{4(n-1)}RPQ)$ , where  $N$  is the vocabulary size of the target language,  $n$  is the length of the  $n$ -gram model,  $R$  is the maximum number of templates applicable to one tree node,  $P$  is the maximum number of variables in a template, and  $Q$  is the number of tree nodes in the syntax tree. The DP algorithm works for most systems in the paper, and only needs to be slightly modified to encode the subtree-based  $n$ -gram model described in Section 3.3.

### 3 Semantics-Based Approaches for Tree-to-string Transducer

#### 3.1 Semantics From PropBank

There are two semantic standards with publicly available training corpus: PropBank (Palmer et al., 2005) and FrameNet (Johnson et al., 2002). PropBank defines a set of semantic roles for the verbs in the Penn TreeBank using numbered roles. The roles are defined individually for each verb. For example, for the verb *disappoint*, the role name *arg1* means *experiencer*, but for verb *wonder*, role name *arg1* means *cause*. FrameNet is motivated by the idea that a certain type of verbs can be gathered together to form a *frame*, and in the same *frame*, a set of semantic roles are defined and shared among the verbs. For example, the verbs *boil*, *bake*, and *steam* will be in frame *apply\_heat*, and they have the semantic roles of *cook*, *food*, and *heating\_instrument*.

```

function DECODE( $T$ )
  for tree node  $v$  of  $T$  in bottom-up order do
    for template  $t$  applicable at  $v$  do
       $\{c_1, c_2, \dots, c_l\} = \text{Match}(v, t)$ ;  $\triangleright$   $\text{Match}(v, t)$ : the descendant tree nodes of  $v$ , which match the variables in
      template  $t$ 
       $\{f_1, f_2, \dots, f_m\} = \text{BinaryCombine}(c_1.sk, c_2.sk, \dots, c_n.sk, t)$ ;  $\triangleright$   $c.sk$ : the stack of tree node  $c$ 
      for  $i=1:m$  do
         $\text{Pr}(f_i) = \prod_{j=1}^l \text{Pr}(\text{In}(c_j, f_i)) \cdot \text{Weight}(t)^\beta \cdot \text{Lang}(v, t, f_i)^\gamma \cdot \text{Penalty}(t)^\alpha$ ;  $\triangleright$   $\text{In}(c_j, f_i)$ : the translation
        candidate of  $c_j$  which is chosen to combine  $f_i$ 
        Add  $(f_i, \text{Pr}(f_i))$  to  $v.sk$ ;
      end for
    end for
    Prune  $v.sk$ ;
  end for
end function

```

Figure 2: Decoding Algorithm

Of these two semantic standards, we choose PropBank over FrameNet for the following reasons:

1. PropBank has simpler semantic definition than FrameNet and thus is easier for automatic labeling.
2. PropBank is built upon Penn TreeBank and more consistent with the automatic parsers, most of which are trained upon Penn TreeBank.
3. PropBank is a larger corpus than FrameNet, which indicates that classifiers trained upon PropBank are expected to have better performance than trained upon FrameNet.
4. There is an English-to-Chinese parallel corpus with human annotated parse trees and semantic roles available which follows PropBank standard, making it possible for us to experiment with gold standard parse trees and semantic roles in a TTS transducer.

The type of semantic standard/corpus is not crucial to the semantics-based TTS transducer algorithm described in this paper. As long as the standard can be used to obtain the set of roles of a verb, it can work with our approaches.

### 3.2 Backing-off Model for semantics-based Tree-to-string Transducer

There are two ways of adding the semantic roles into the syntax trees: one is to generate a set of *general*

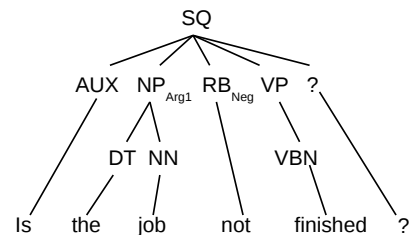


Figure 3: A tree example with refined labels

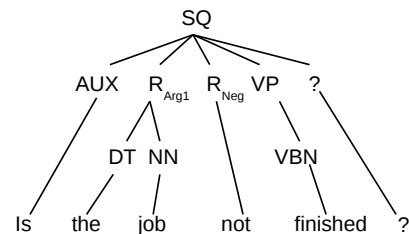


Figure 4: A tree example with general labels

*labels* by replacing the syntax labels with the semantic roles; the other way is to generate a set of *refined labels* by combining the syntax labels with the semantic roles. The motivation of doing this is that the subtrees, with different roles attached, may have different re-ordering in the TTS transducing. Figure 4 and Figure 3 shows the examples of the tree fragments with *general/refined* labels.

For the approach of using refined labels, when a semantic role-based subtree cannot be matched by any semantic template, it is necessary to see if the subtree, after removing all the semantic roles, can be matched by the syntax-based templates. This backing-off model for semantics-based TTS transducer can be integrated into the decoder by defining

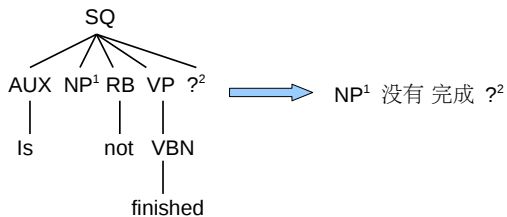


Figure 5: An example of the skeleton template

template applicable at  $v$  in Figure 2 as:

1. The semantic templates matching subtrees rooted at  $v$ , or
2. The syntactic templates matching subtrees rooted at  $v$  which are not matched by the semantic templates.

Since the semantic templates with refined labels could be sparse in the training data, it is useful to smooth their probabilities with probabilities of syntax-based templates using interpolation, as described below:

$$Pr(R_{refine'}) = \frac{Pr(R_{refine}) \times \theta + Pr(R_{syntax})}{1 + \theta}$$

The interpolation weight  $\theta$  can be tuned based on the development data. When a tree node is labeled with different semantic roles for different verbs, those semantic roles will not be used.

### 3.3 Skeleton Templates

One drawback of the approach using *general/refined* tree labels is that the bond of the verb and its roles is not used. Further adding verbs into the tree labels is a way to use such bonds but might cause the templates to be too sparse to be useful. Here we propose another way called *Skeleton Template Extraction* to utilize such information. Skeleton here denotes, for a certain verb, the minimum subtree containing the verb and all its identified semantic roles. The skeleton templates thus address the re-ordering of the sentence skeleton centered by the verb. Figure 5 shows an example of a skeleton template. The skeleton templates are not necessarily (and usually not) the minimum templates which can be extracted using

```

function EXTRACT_SKELETON( $T$ )
  for verb  $v$  in a sentence  $T$  do
     $lca$  = the lowest common ancestor of  $v$  and all
    its roles
    while  $lca \neq \text{NULL}$  AND  $lca$  is not a frontier
    node do
       $lca$  =  $lca$ 's parent
    end while
    EXPAND( $lca$ ,  $v$ )
  end for
end function

function EXPAND( $t$ ,  $v$ )
  MARK  $t$ ;  $\triangleright$  tree node  $t$  will be in the extracted
  template
  for all children  $c$  of  $t$  do
    if  $c$  is not a frontier node OR  $c$  or  $c$ 's descen-
    dants are  $v$  or  $v$ 's roles then
      EXPAND( $c$ );
    end if
  end for
end function

```

Figure 6: Algorithm for extracting skeleton templates

GHKM (Galley et al., 2004). To get the skeleton templates, we first find the lowest common ancestor of a verb and all its roles. If the lowest common ancestor is not a frontier tree node (Galley et al., 2004), we ascend until a frontier node is met. Down from the lowest frontier common ancestor, the minimum TTS template containing the verb and all its roles is extracted as the skeleton template. The formal description of the template extraction algorithm is in Figure 6.

## 4 Experiments

### 4.1 Data Preparation And Evaluation

To obtain the syntax trees and semantic roles for the TTS transducer, an automatic parsing and semantic role labeling system is required to pre-process the input sentences. We use Charniak (2000)'s parser and an in-house maximum entropy classifier with features following Xue and Palmer (2004) and Ward et al. (2004) for role labeling, and they are trained on sections 2–21 of Penn TreeBank and PropBank respectively. The parallel corpora used to train the translation model and language model have two parts: one is 73,597 English-Chinese sentence pairs in FBIS corpus, and the other is a small bilingual

	Trained on auto FBIS		Trained on auto/gold SBC	
	avg BLEU	95%	avg BLEU	95%
Tested on auto SBC	6.78	5.57-8.39	5.68	4.66-7.09
Tested on gold SBC	7.09	5.88-8.87	9.76	8.60-11.89

Table 1: Comparison of MT performance with automatic/gold-standard parses.

corpus with gold-standard parses and semantic roles obtained from the University of Colorado. The small bilingual corpus is aligned at the article level, but not at the sentence level. To be able to use it in the MT experiments, the article pairs are further divided and aligned to form 3,200 English-Chinese sentence pairs. For convenience, SBC (small bilingual corpus) is used to denote this corpus with gold standard parses and semantic roles.

The evaluation of the parsing and the SRL follows the standard bracket-based F-score and role-based F-score respectively. The weights of the MT components including the interpolation weight of the semantics-based TTS transducer are tuned based on the selected development set using a grid-based line search. The metric used to evaluate the MT system is the word-based BLEU-4 (Papineni et al., 2002), based on the single reference of the selected test set from FBIS or SBC. The templates of TTS transducer are normalized using the first-level expansion-based method described in ?).

#### 4.2 Performance of the Automatic Parsing and Semantic Role Labeling

A lot of work has been done to improve the performance of statistical parsing and semantic role labeling (Collins, 2000; Charniak, 2000; Xue and Palmer, 2004; Ward et al., 2004; Toutanova et al., 2005). Charniak (2000)’s parser represents the state-of-art and achieves a F-score of 89.5% in section 23, the standard test set of Penn TreeBank, with sentences of length 100 and less. The best SRL results on PropBank’s standard test set is around 91.2% in F-score with gold-standard parses (Toutanova et al., 2005), and the performance drops to around 80% if the automatic parses are used (Toutanova et al., 2005). Our in-house built SRL system got an F-score of 88.3% with the gold-standard parses, which is not as good as, but close to, the state-of-art. It seems that both parsing and SRL have achieved decent performance based on Penn Tree-

	$\leq 0.1$	0.2-0.5	0.5-0.8	0.8-0.9	0.9-1
FBIS	95.52%	2.51%	1.66%	0.18%	0.13%
SBC	93.30%	3.87%	2.37%	0.38%	0.08%

Table 3: Distribution of L1 scores of the refined templates and their syntactic counterparts.

Bank, but will they perform as well in the non-Penn-TreeBank data? We will experiment with automatically parsed and labeled FBIS corpus, so it is important to know how reliable the automatically generated syntax trees and semantic roles are. To find out the answer, SBC is used as the test set, and the F-scores of automatic parsing and SRL (with automatic parses) are 74.0% and 69.1% respectively. SBC is actually the same type of news-wire corpus as Penn TreeBank, though from Chinese news agencies. We can see a big decrease in the performance in both parsing and SRL. It shows that for non-Penn-TreeBank data (including our MT experiments data), the statistical parser and SRL system might not work as well as they do for the Penn TreeBank data.

#### 4.3 Automatic Parsing vs. Gold Standard Parsing

We have shown the poor performance of automatic parsing on non-Penn-TreeBank data, but is the parsing quality really going to affect the performance of a syntax-based MT system such as a TTS transducer? To find out the answer, experiments are conducted with gold-standard/automatic parses. In the first experiment, the FBIS corpus with automatic parses is used as the training set, and the SBC with automatic/gold-standard parses is used as the test set. To make the results comparable with the  $n$ -fold cross validation, SBC is divided into 8 folds of 200 development/test sentences and the averaged BLEU score of the 8 folds, as well as the 95% confidence interval based on 2,000 times re-sampling (Koehn, 2004) of the 8 folds, are shown in Table 1. We can see that trained on automatic parses, the TTS

	Auto FBIS		Auto FBIS (SRL $\geq$ 0.9)		Gold SBC	
	avg BLEU	95%	avg BLEU	95%	avg BLEU	95%
Syntax Labels	11.17	10.19-12.07	–	–	9.76	8.60-11.89
PropBank General Labels	11.06	10.13-12.08	11.00	10.02-12.13	9.85	8.63-11.97
PropBank Refined Labels	11.18	10.20-12.13	11.04	10.10-12.08	9.92	8.66-11.90
Semantic Refined Labels	11.06	10.04-11.99	10.92	10.01-12.02	9.67	8.55-11.67

Table 2: Comparison of MT performance with syntax/semantic trees.

transducer works little bit better on the test set with gold-standard parse than the test set with automatic parses.

The second experiment does cross-validation with SBC so that we can both train and test on gold-standard parses. Table 1 shows that there is a significant improvement when the TTS transducer is trained and tested upon gold-standard parses vs. trained and tested upon automatic parses. It indicates that improvements in statistical parsing are likely to lead to the improvement of TTS transducer. Since the two sets of experiments use the same test set, their results are also comparable. Comparing the first row of Table 1, we can see that trained on automatic parses, FBIS beats SBC indicating that more data wins. If we compare the second row, smaller SBC with gold-standard parses beats automatically parsed FBIS, indicating that higher quality wins over larger amount.

#### 4.4 Syntax Trees vs. Semantic Trees

?) reported preliminary results by using the semantics-based refined/general labels described in section 3.2, here more reliable results based on N-fold cross validation will be presented. For the approach of using refined labels, the back-off model with interpolation described in Section 3.2 is used. To compare the results based on different quality of SRL, 3 sets of data are used including automatically parsed and labeled FBIS, automatically parsed and labeled FBIS which only keeps the semantic roles of probability higher than 0.9 based on the SRL classifier, and SBC with gold-standard parses and semantic roles. The averaged scores over 10-folds and the 95% confidence intervals based on 2,000 times resampling are shown in Table 2. We can see that with gold-standard SBC, the semantic labels are little bit better than the syntax labels, and with automatically parsed and labeled FBIS, syntax labels are little bit

better than semantic labels. Overall their difference is very small and not statistically significant. Retaining only those semantic role labels classified with high confidence by the SRL system does not improve the semantics-based TTS transducer.

Role names in PropBank have different meanings for different verbs, and such information is not used in our semantics-based approach. To see if this is the reason that the semantic trees are not working better, we use the role mapping table associated with PropBank to map the core role names (e.g., arg0, arg1) to a set of 58 thematic roles (e.g. Agent, Patient) based on verbs and their most common senses. The results are shown in the *Semantic Refined Labels* row of Table 2, but unfortunately this doesn't make much difference either. It seems that replacing the syntax labels with semantics-based labels does not change the output of TTS transducer very much, and we cannot help but wonder: do the semantic roles really make any difference for a TTS transducer?

##### 4.4.1 Are Syntax/semantics-based TTS Templates Really Different?

As mentioned in Section 3.2, the motivation of replacing tree label with semantics-based labels is that the subtrees, with different roles attached, may have different re-ordering in the TTS transducing. The similar results we got on semantic/syntactic TTS transducer lead us question the validity of this assumption. To compare the difference of semantic templates and their syntactic counterparts, we can compare the distribution of the RHS given the LHS with and without the semantic roles. Here we consider the RHS distributions as vectors, and compare them based on L1 distance. To make the scores fall in between 0 and 1, the L1 scores are divided by 2. Table 3 shows the distribution of the L1 scores of the *refined* templates and their syntax counterparts, extracted from both automatically processed FBIS and

	Gold SBC			Auto FBIS		
	# templates	avg BLEU	95%	# templates	avg BLEU	95%
Minimum Templates	30k	9.76	8.60-11.89	350k	11.17	10.19-12.07
+ Long Templates	60k	9.94	8.70-12.06	780k	11.40	10.38-12.29
+ Skeleton Templates	35k	10.06	9.04-11.83	415k	11.16	10.19-12.15

Table 4: Performance of straight long/skeleton templates.

the gold-standard SBC. The statistics based on FBIS and SBC are similar and show that most templates with refined labels are actually very similar to their syntactic counterparts, which, we speculate, is why the semantic roles do not improve the syntax-based TTS transducer.

#### 4.5 Skeleton Templates vs. Straight Long Templates

Using the skeleton templates is a way to extend the template set beyond the minimum templates defined in GHKM. Usually having more templates in the system will not hurt the performance, so to see if the skeleton templates are really better than the randomly chosen long templates, another approach, which extracts the long templates all the way down to depth 6, is used to compare with it. Table 4 shows the BLEU-4 scores based on both automatically processed FBIS and gold-standard SBC as well as the resulting number of templates for the two approaches. We can see from the first column that with gold-standard SBC, skeleton templates, although they still do not have significant improvement over the minimum templates, outperform the straight long templates with a much smaller template set. It indicates that extracting skeleton templates is a good way to select long templates. Unfortunately, for the automatically parsed and labeled FBIS data, adding in skeleton templates doesn't improve the performance over the minimum templates and not as good as adding in the straight long templates. We speculate it is the high error rate of the parsing/semantic role labeling that makes the extracted skeleton templates not as accurate as those extracted based on gold-standard parses and semantic roles.

## 5 Discussion And Future Work

We have shown that adding semantic roles into the syntax tree labels is not a good way of utilizing the semantic features. One alternative might be a two-layer translation model: the first layer is the translation based on skeleton templates, the second layer is the translation of the leaves of the skeleton templates, i.e., verbs and their roles. The two-layer model can become *more semantic* by using a semantics-based flat tree instead of a full syntax tree for the skeleton templates. A semantics-based flat tree is an one-level tree formed by ignoring the internal syntax structure of the original syntax subtree and only keeping the root and the leaves. This leads to a verb-centered hybrid system featuring both the semantics-based hierarchical reordering and the phrase-based translation. Semantic roles can also be used as features in discriminative models for MT. For example, they might be useful to help select the proper translation for source words.

## 6 Conclusion

This paper first compares the performance of a TTS transducer with automatically generated and gold-standard parse trees, and finds that improving the parsing quality can lead to significant improvement in MT performance. Then the approach of adding semantic roles into the syntax tree labels is shown not to improve the TTS transducer and reason is the syntax/semantics-based templates are not very different in terms of the RHS distribution. We finally demonstrate some promise of the skeleton templates extraction approach and discuss how to make a semantic flat-tree-based transducer along the direction.

## References

- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL-01*, pages 132–139.
- Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*, pages 175–182, Stanford, California.
- Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What’s in a translation rule? In *Proceedings of NAACL-04*, pages 273–280.
- Michel Galley, Jonathan Graehl, Kevin Knight, Daniel Marcu, Steve DeNeefe, Wei Wang, and Ignacio Thayer. 2006. Scalable inference and training of context-rich syntactic translation models. In *Proceedings of COLING/ACL-06*, pages 961–968, July.
- Jonathan Graehl and Kevin Knight. 2004. Training tree transducers. In *Proceedings of NAACL-04*.
- Liang Huang, Kevin Knight, and Aravind Joshi. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of the 7th Biennial Conference of the Association for Machine Translation in the Americas (AMTA)*, Boston, MA.
- Christopher R. Johnson, Charles J. Fillmore, Miriam R. L. Petruck, Collin F. Baker, Michael Ellsworth, Josef Ruppenhofer, and Esther J. Wood. 2002. FrameNet: Theory and practice. Version 1.0, <http://www.icsi.berkeley.edu/framenet/>.
- Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, pages 388–395, Barcelona, Spain, July.
- Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of COLING/ACL-06*, Sydney, Australia, July.
- J. May and K. Knight. 2007. Syntactic re-alignment models for machine translation. In *Proceedings of EMNLP*.
- Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational Linguistics*, 30(4):417–449.
- Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL-02*.
- William C. Rounds. 1970. Mappings and grammars on trees. *Mathematical Systems Theory*, 4(3):257–287.
- J. W. Thatcher. 1970. Generalized<sup>2</sup> sequential machine maps. *J. Comput. System Sci.*, 4:339–367.
- Kristina Toutanova, Aria Haghighi, and Christopher Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of ACL-05*, pages 589–596.
- Wayne Ward, Kadri Hacioglu, James Martin, , and Dan Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proceedings of EMNLP*.
- Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.
- Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP*.
- Hao Zhang, Liang Huang, Daniel Gildea, and Kevin Knight. 2006. Synchronous binarization for machine translation. In *Proceedings of NAACL-06*, pages 256–263, New York, NY.