Semantic Role Features for Machine Translation

Ding Liu
Department of Computer Science
University of Rochester

Daniel Gildea
Department of Computer Science
University of Rochester

Abstract

We propose semantic role features for a Tree-to-String transducer to model the reordering/deletion of source-side semantic roles. These semantic features, as well as the Tree-to-String templates, are trained based on a conditional log-linear model and are shown to significantly outperform systems trained based on Max-Likelihood and EM. We also show significant improvement in sentence fluency by using the semantic role features in the log-linear model, based on manual evaluation.

1 Introduction

Syntax-based statistical machine translation (SSMT) has achieved significant progress during recent years (Galley et al., 2006; May and Knight, 2007; Liu et al., 2006; Huang et al., 2006), showing that deep linguistic knowledge, if used properly, can improve MT performance. Semantics-based SMT, as a natural extension to SSMT, has begun to receive more attention from researchers (Liu and Gildea, 2008; Wu and Fung, 2009). Semantic structures have two major advantages over syntactic structures in terms of helping machine translation. First of all, semantic roles tend to agree better between two languages than syntactic constituents (Fung et al., 2006). This property motivates the approach of using the consistency of semantic roles to select MT outputs (Wu and Fung, 2009). Secondly, the set of semantic roles of a predicate models the *skeleton* of a sentence, which is crucial to the readability of MT output. By skeleton, we mean the main structure of a sentence including the verbs and their arguments. In spite of the theoretical potential of the semantic roles, there has not been much success in using them to improve SMT systems.

Liu and Gildea (2008) proposed a semantic role based Tree-to-String (TTS) transducer by adding semantic roles to the TTS templates. Their approach did not differentiate the semantic roles of different predicates, and did not always improve the TTS transducer's performance. Wu and Fung (2009) took the output of a phrase-based SMT system Moses (Koehn et al., 2007), and kept permuting the semantic roles of the MT output until they best matched the semantic roles in the source sentence. This approach shows the positive effect of applying semantic role constraints, but it requires re-tagging semantic roles for every permuted MT output and does not scale well to longer sentences.

This paper explores ways of tightly integrating semantic role features (SRFs) into an MT system, rather than using them in post-processing or n-best re-ranking. Semantic role labeling (SRL) systems usually use sentence-wide features (Xue and Palmer, 2004; Pradhan et al., 2004; Toutanova et al., 2005); thus it is difficult to compute target-side semantic roles incrementally during decoding. Noticing that the source side semantic roles are easy to compute, we apply a compromise approach, where the target side semantic roles are generated by projecting the source side semantic roles using the word alignments between the source and target sentences. Since this approach does not perform true SRL on the target string, it cannot fully evaluate whether the source and target semantic structures are consistent. However, the approach does capture the semantic-level re-ordering of the sentences. We assume here that the MT system is capable of providing word alignment (or equivalent) information during decoding, which is generally true for current statistical MT systems.

Specifically, two types of semantic role features are proposed in this paper: a semantic role reordering feature designed to capture the skeleton-level permutation, and a semantic role deletion fea-

ture designed to penalize missing semantic roles in the target sentence. To use these features during decoding, we need to keep track of the semantic role sequences (SRS) for partial translations, which can be generated based on the source-side semantic role sequence and the corresponding word alignments. Since the SRL system and the MT system are separate, a translation rule (e.g., a phrase pair in phrase-based SMT) could cover two partial source-side semantic roles. In such cases partial SRSs must be recorded in such a way that they can be combined later with other partial SRSs. Dealing with this problem will increase the complexity of the decoding algorithm. Fortunately, Tree-to-String transducer based MT systems (Liu et al., 2006; Huang et al., 2006) can avoid this problem by using the same syntax tree for both SRL and MT. Such an arrangement guarantees that a TTS template either covers parts of one source-side semantic role, or a few complete semantic roles. This advantage motivates us to use a TTS transducer as the MT system with which to demonstrate the use of the proposed semantic role features. Since it is hard to design a generative model to combine both the semantic role features and the TTS templates, we use a log-linear model to estimate the feature weights, by maximizing the conditional probabilities of the target strings given the source syntax trees. The log-linear model with latent variables has been discussed by Blunsom et al. (2008); we apply this technique to combine the TTS templates and the semantic role features.

The remainder of the paper is organized as follows: Section 2 describes the semantic role features proposed for machine translation; Section 3 describes how semantic role features are used and trained in a TTS transducer; Section 4 presents the experimental results; and Section 5 gives the conclusion.

2 Semantic Role Features for Machine Translation

2.1 Defining Semantic Roles

There are two semantic standards with publicly available training data: 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. These roles are defined individually for each verb. For example, for the verb *disappoint*, the role name *arg1* means *experiencer*, but for the 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 is 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*. Of these two semantic standards, we choose PropBank over FrameNet for the following reasons:

1. PropBank has a simpler semantic definition than FrameNet and thus is easier for automatic labeling.

2. PropBank is built upon the Penn TreeBank and is more consistent with statistical parsers, most of which are trained on the Penn TreeBank.

3. PropBank is a larger corpus than FrameNet.

Note that the semantic standard/corpus is not crucial in this paper. Any training corpus that can be used to automatically obtain the set of semantic roles of a verb could be used in our approach.

2.2 Semantic Role Features

Ideally, we want to use features based on the true semantic roles of the MT candidates. Considering there is no efficient way of integrating SRL and MT, accurate target-side semantic roles can only be used in post-processing and re-ranking the MT outputs, where a limited number of MT candidates are considered. On the other hand, it is much easier to obtain reliable semantic roles for the source sentences. This paper uses a compromise approach, where the target-side semantic roles are projected from the source-side semantic roles using the word alignment derived from the translation process. More specifically, we define two types of semantic role features:

1. **Semantic Role Re-ordering (SRR)** This feature describes re-ordering of the source-side

semantic roles (including the predicate) in the target side. It takes the following form:

$$SrcPred : SrcRole_1, ..., SrcRole_n \Rightarrow TarRole_1, ..., TarRole_n$$

where $SrcPred$ and $SrcRole$ denotes the central verb and semantic roles in the source side, and $TarRole$ denotes the target-side roles. The source/target SRSs do not need be continuous, but there should be a one-to-one alignment between the roles in the two sides. Compared to the general re-ordering models used in statistical MT systems, this type of feature is capable of modeling skeleton-level re-ordering, which is crucial to the fluency of MT output. Because a predicate can have different semantic role sequences in different voices, *passive/active* are tagged for each occurrence of the verbs based on their POS and preceding words. Figure 1 shows examples of the feature SRR.

2. **Deleted Roles (DR)** are the individual source-side semantic roles which are deleted in the MT outputs, taking the form of:

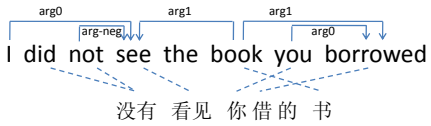
$$SrcPred : SrcRole \Rightarrow deleted$$

DR is meant to penalize the deletion of the semantic roles. Though most statistical MT systems have penalties for word deletion, it is still useful to make separate features for the deletion of semantic roles, which is considered more harmful than the deletion of non-core components (e.g., modifiers) and deserves more serious penalty. Examples of the deletion features can be found in Figure 1.

Both types of features can be made non-lexicalized by removing the actual verb but retaining its voice information in the features. Non-lexicalized features are used in the system to alleviate the problem of sparse verbs.

3 Using Semantic Role Features in Machine Translation

This section describes how to use the proposed semantic role features in a Tree-to-String transducer,



SRR:
see-active: arg-neg verb ➔ arg-neg verb
borrowed-active: arg1 arg0 ➔ arg0 arg1
borrowed-active: arg1 verb ➔ verb arg1
borrowed-active: arg0 verb ➔ arg0 verb
borrowed-active: arg1 arg0 verb ➔ arg0 verb arg1
DR:
see-active: arg0 ➔ deleted

Figure 1: Examples of the semantic role features

assuming that the semantic roles have been tagged for the source sentences. We first briefly describe the basic Tree-to-String translation model used in our experiments, and then describe how to modify it to incorporate the semantic role features.

3.1 Basic Tree-to-String Transducer

A Tree-to-String transducer receives a syntax tree as its input and, by recursively applying TTS templates, generates the target string. A TTS template is composed of a left-hand side (LHS) and a right-hand side (RHS), where the LHS is a subtree pattern and the RHS is a sequence of 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). For a given derivation (decomposition into templates) 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.

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

Here, S denotes the target sentence, T denotes the source syntax tree, and D^* denotes the derivation of T. In addition to the translation model, the

function DECODE(T)
  for tree node v of T in bottom-up order **do**
    for template t applicable at v **do**

      $\{c_1, c_2\}$=match(v, t);
      $s.leftw$ = $c_1.leftw$;
      $s.rightw$ = $c_2.rightw$;
      $s.val$ = $c_1.val \times c_2.val$;
      $s.val$ $\times$= $Pr(t)$;
      $s.val$ $\times$= $Pr(c_2.leftw|c_1.rightw)$;
      add s to v's beam;

Figure 2: Decoding algorithm for the standard Tree-to-String transducer. $leftw$/$rightw$ denote the left/right boundary word of s. c_1, c_2 denote the descendants of v, ordered based on RHS of t.

TTS system includes a trigram language model, a deletion penalty, and an insertion bonus. The bottom-up decoding algorithm for the TTS transducer is sketched in Figure 2. To incorporate the n-gram language model, states in the algorithm denote a tree node's best translations with different left and right boundary words. We use standard beam-pruning to narrow the search space. To simplify the description, we assume in Figure 2 that a bigram language model is used and all the TTS templates are binarized. It is straightforward to generalize the algorithm for larger n-gram models and TTS templates with any number of children in the bottom using target-side binarized combination (Huang et al., 2006).

3.2 Modified Tree-to-String Transducer with Semantic Role Features

Semantic role features can be used as an auxiliary translation model in the TTS transducer, which focuses more on the skeleton-level permutation. The model score, depending on not only the input source tree and the derivation of the tree, but also the semantic roles of the source tree, can be formulated as:

$$Pr(S \mid T, D^*) = \prod_{f \in F(S, T.role, D^*)} Pr(f)$$

where T denotes the source syntax tree with semantic roles, $T.role$ denotes the semantic role sequence in the source side and $F(S.role, T.role, D^*)$ denotes the set of defined semantic role features over $T.role$ and the target side semantic role sequence $S.role$. Note that given $T.role$ and the derivation D^*, $S.role$ can

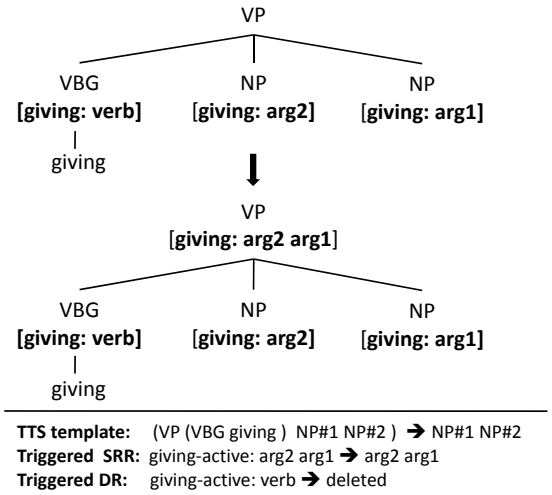


Figure 3: An example showing the combination of the semantic role sequences of the states. Above/middle is the state information before/after applying the TTS template, and bottom is the used TTS template and the triggered SRFs during the combination.

be easily derived. Now we show how to incorporate the two types of semantic role features into a TTS transducer. To use the semantic role re-ordering feature SRR, the states in the decoding algorithm need to be expanded to encode the target-side SRSs. The SRSs are initially attached to the translation states of the source tree con-

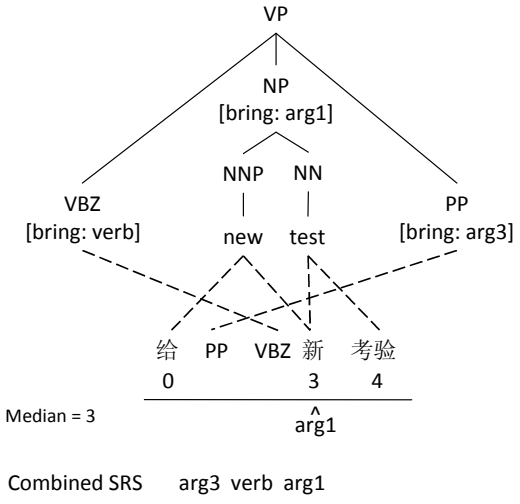


Figure 4: An example showing how to compute the target side position of a semantic role by using the median of its aligning points.

stituents which are labeled as semantic roles for some predicate. These semantic roles are then accumulated with re-ordering and deletion operations specified by the TTS templates as the decoding process goes bottom-up. Figure 5 shows the decoding algorithm incorporating the SRR features. The model component corresponding to the feature SRR is computed when combining two translation states. I.e., the probabilities of the SRR features composed based on the semantic roles of the two combining states will be added into the combined state. See Figure 3 for examples. The theoretical upper bound of the decoding complexity is $O(NM^{4(n-1)}R(\sum_{i=0}^{C}\frac{C!}{i!})^V)$, where N is the number of nodes in the source syntax tree, M is the vocabulary size of the target language, n is the order of the n-gram language model, R is the maximum number of TTS templates which can be matched at a tree node, C is the maximum number of roles of a verb, and V is the maximum number of verbs in a sentence. In this formula, $\sum_{i=0}^{C}\frac{C!}{i!}$ is the number of role sequences obtained by first choosing i out of C possible roles and then permuting the i roles. This theoretical upper bound is not reached in practice, because the number of possible TTS templates applicable at a tree node is very limited. Furthermore, since we apply beam pruning at each tree node, the running time is controlled by the beam size, and is linear in the size of the tree.

The re-ordering of the semantic roles from source to target is computed for each TTS template as part of the template extraction process, using the word-level alignments between the LHS/RHS of the TTS template (e.g., Figure 3). This is usually straightforward, with the exception of the case where the words that are aligned to a particular role's span in the source side are not continuous in the target side, as shown in Figure 4. Since we are primarily interested in the relative order of the semantic roles, we approximate each semantic role's target side position by the median of the word positions that is aligned to. If more than one semantic role is mapped to the same position in the target side, their source side order will be used as their target side order, i.e., monotonic translation is assumed for those semantic roles. Figure 4 shows an example of calculating the target side

function DECODE(T)
  for tree node v of T in bottom-up order **do**
    for template t applicable at v **do**
      $\{c_1, c_2\}$=match(v, t);
      $s.leftw$ = $c_1.leftw$;
      $s.rightw$ = $c_2.rightw$;
      $s.role$ = concatenate($c_1.role$, $c_2.role$);
      if v is a semantic role **then**
        set *s.role* to *v.role*;
      $s.val$ = $c_1.val \times c_2.val$;
      $s.val \times= Pr(t)$;
      $s.val \times= Pr(c_2.leftw|c_1.rightw)$;
    ▷ Compute the probabilities associated with semantic roles
      $s.val \times= \prod_{f \in Sema(c_1.role, c_2.role, t)} Pr(f)$;
      add s to v's beam;

Figure 5: Decoding algorithm using semantic role features. $Sema(c_1.role, c_2.role, t)$ denotes the triggered semantic role features when combining two children states, and examples can be found in Figure 3.

SRS based on a complicated TTS template. The word alignments in the TTS templates are also used to compute the deletion feature DR. Whenever a semantic role is deleted in a TTS template's RHS, the corresponding deletion penalty will be applied.

3.3 Training

We describe two alternative methods for training the weights for the model's features, including both the individual TTS templates and the semantic role features. The first method maximizes data likelihood as is standard in EM, while the second method maximizes conditional likelihood for a log-linear model following Blunsom et al. (2008).

3.3.1 Maximizing Data Likelihood

The standard way to train a TTS translation model is to extract the minimum TTS templates using GHKM (Galley et al., 2004), and then normalize the frequency of the extracted TTS templates (Galley et al., 2004; Galley et al., 2006; Liu et al., 2006; Huang et al., 2006). The probability of the semantic features SRR and DR can be computed similarly, given that SRR and DR can be derived from the paired source/target sentences and the word alignments between them. We refer to this model as max-likelihood training and normalize the counts of TTS templates and semantic features based on their roots and predicates respectively.

We wish to overcome noisy alignments from GIZA++ and learn better TTS rule probabilities by re-aligning the data using EM within the TTS

E-step:

for all pair of syntax tree T and target string S **do**
  for all TTS Template t, semantic features f **do**
    $EC(t)$ += $\frac{\sum_{D:t\in D} Pr(S,T,D)}{\sum_{D'} Pr(S,T,D')}$;
    $EC(f)$ += $\frac{\sum_{D:f\in D} Pr(S,T,D)}{\sum_{D'} Pr(S,T,D')}$;

M-step:

for all TTS Template t, semantic features f **do**
  $Pr(t) = \frac{EC(t)}{\sum_{t':t'.root=t.root} EC(t')}$;
  $Pr(f) = \frac{EC(f)}{\sum_{f':f'.predicate=t.predicate} EC(f')}$;

Figure 6: EM Algorithm For Estimating TTS Templates and Semantic Features

function COMPUTE_PARTITION(T)
  for tree node v of T in bottom-up order **do**
    for template t applicable at v **do**
      for $\{s_1, s_2\}$=Match(v, t) **do**
        $s.sum$ += $s_1.sum \times s_2.sum \times exp(\lambda_t + \sum_{f\in Sema(s_1,s_2,t)} \lambda_f)$;
        $s.role$ = concatenate($s_1.role$, $s_2.role$);
        add s to v;
  for state s in $root$ **do** res += $s.sum$;
  return res;

Figure 7: Computing the partition function of the conditional probability $Pr(S|T)$. $Sema(s_1, s_2, t)$ denotes all the semantic role features generated by combining s_1 and s_2 using t.

framework (May and Knight, 2007). We can estimate the expected counts of the TTS templates and the semantic features by formulating the probability of a pair of source tree and target string as:

$$\sum_D Pr(S,T,D) = \sum_D \left(\prod_{t\in D} Pr(t) \prod_{f\in F(S,T.role,D)} Pr(f) \right)$$

Though the above formulation, which makes the total probability of all the pairs of trees and strings less than 1, is not a strict generative model, we can still use the EM algorithm (Dempster et al., 1977) to estimate the probability of the TTS templates and the semantic features, as shown in Figure 6.

The difficult part of the EM algorithm is the *E-step*, which computes the expected counts of the TTS templates and the semantic features by summing over all possible derivations of the source trees and target strings. The standard inside-outside algorithm (Graehl and Knight, 2004) can be used to compute the expected counts of the TTS templates. Similar to the modification made in the TTS decoder, we can add the target-side semantic role sequence to the dynamic programming states of the inside-outside algorithm to compute the expected counts of the semantic features. This way each state (associated with a source tree node) represents a target side span and the partial SRSs. To speed up the training, a beam is created for each target span and only the top rated SRSs in the beam are kept.

3.3.2 Maximizing Conditional Likelihood

A log-linear model is another way to combine the TTS templates and the semantic features together. Considering that the way the semantic role features are defined makes it impossible to design a sound generative model to incorporate these features, a log-linear model is also a theoretically better choice than the EM algorithm. If we directly translate the EM algorithm into the log-linear model, the problem becomes maximizing the data likelihood represented by feature weights instead of feature probabilities:

$$Pr(S,T) = \frac{\sum_D \exp \sum_i \lambda_i f_i(S,T,D)}{\sum_{S',T'} \sum_{D'} \exp \sum_i \lambda_i f_i(S',T',D')}$$

where the features f include both the TTS templates and the semantic role features. The numerator in the formula above can be computed using the same dynamic programming algorithm used to compute the expected counts in the EM algorithm. However, the partition function (denominator) requires summing over all possible source trees and target strings, and is infeasible to compute. Instead of approximating the partition function using methods such as sampling, we change the objective function from the data likelihood to the conditional likelihood:

$$Pr(S \mid T) = \frac{\sum_D \exp \sum_i \lambda_i f_i(S,T,D)}{\sum_{S'\in all(T)} \sum_{D'} \exp \sum_i \lambda_i f_i(S',T,D')}$$

where $all(T)$ denotes all the possible target strings which can be generated from the source tree T. Given a set of TTS templates, the new partition function can be efficiently computed using the dynamic programming algorithm shown in Figure 7. Again, to simplify the illustration, only binary TTS templates are used. Using the conditional probability as the objective function not only reduces the computational cost, but also corresponds better to the TTS decoder, where the best MT output is

selected only among the possible candidates which can be generated from the input source tree using TTS templates.

The derivative of the logarithm of the objective function (over the entire training corpus) w.r.t. a feature weight can be computed as:

$$\frac{\partial \log \prod_{S,T} Pr(S \mid T)}{\partial \lambda_i} = \sum_{S,T} \{EC_{D|S,T}(f_i) - EC_{S'|T}(f_i)\}$$

where $EC_{D|S,T}(f_i)$, the expected count of a feature over all derivations given a pair of tree and string, can be computed using the modified inside-outside algorithm described in Section 3.2, and $EC_{S'|T}(f_i)$, the expected count of a feature over all possible target strings given the source tree, can be computed in a similar way to the partition function described in Figure 7. With the objective function and its derivatives, a variety of optimization methods can be used to obtain the best feature weights; we use LBFGS (Zhu et al., 1994) in our experiments. To prevent the model from overfitting the training data, a weighted Gaussian prior is used with the objective function. The variance of the Gaussian prior is tuned based on the development set.

4 Experiments

We train an English-to-Chinese translation system using the FBIS corpus, where 73,597 sentence pairs are selected as the training data, and 500 sentence pairs with no more than 25 words on the Chinese side are selected for both the development and test data.[1] Charniak (2000)'s parser, trained on the Penn Treebank, is used to generate the English syntax trees. To compute the semantic roles for the source trees, we use an in-house max-ent classifier with features following Xue and Palmer (2004) and Pradhan et al. (2004). The semantic role labeler is trained and tuned based on sections 2–21 and section 24 of PropBank respectively. The standard role-based F-score of our semantic role labeler is 88.70%. Modified Kneser-Ney trigram models are trained using SRILM (Stolcke, 2002) on the Chinese portion of the training data. The model (n-gram language model, TTS templates, SRR, DR) weights of the transducer are tuned based on the development set using a grid-based line search, and the translation results are evaluated based on a single Chinese reference using BLEU-4 (Papineni et al., 2002). Huang et al. (2006) used character-based BLEU as a way of normalizing inconsistent Chinese word segmentation, but we avoid this problem as the training, development, and test data are from the same source.

The baseline system in our experiments uses the TTS templates generated by using GHKM and the union of the two single-direction alignments generated by GIZA++. Unioning the two single-direction alignments yields better performance for the SSMT systems using TTS templates (Fossum et al., 2008) than the two single-direction alignments and the heuristic diagonal combination (Koehn et al., 2003). The two single-direction word alignments as well as the union are used to generate the initial TTS template set for both the EM algorithm and the log-linear model. The initial TTS templates' probabilities/weights are set to their normalized counts based on the root of the TTS template (Galley et al., 2006). To test semantic role features, their initial weights are set to their normalized counts for the EM algorithm and to 0 for the log-linear model. The performance of these systems is shown in Table 1. We can see that the EM algorithm, based only on TTS templates, is slightly better than the baseline system. Adding semantic role features to the EM algorithm actually hurts the performance, which is not surprising since the combination of the TTS templates and semantic role features does not yield a sound generative model. The log-linear model based on TTS templates achieves significantly better results than both the baseline system and the EM algorithm. Both improvements are significant at $p < 0.05$ based on 2000 iterations of paired bootstrap resampling of the test set (Koehn, 2004).

Adding semantic role features to the log-linear model further improves the BLEU score. One problem in our approach is the sparseness of the verbs, which makes it difficult for the log-linear model to tune the lexicalized semantic role features. One way to alleviate this problem is to make features based on verb classes. We first tried using the verb

[1] The total 74,597 sentence pairs used in experiments are those in the FBIS corpus whose English part can be parsed using Charniak (2000)'s parser.

	TTS Templates	+ SRF	+ Verb Class
Union	15.6	–	–
EM	15.9	15.5	15.6
Log-linear	17.1	17.4	17.6

Table 1: BLEU-4 scores of different systems

	equal	better	worse
With SRF vs. W/O SRF	72%	20.2%	7.8%

Table 2: Distribution of the sentences where the semantic role features give no/positive/negative impact to the sentence fluency in terms of the completeness and ordering of the semantic roles.

classes in VerbNet (Dang et al., 1998). Unfortunately, VerbNet only covers about 34% of the verb tokens in our training corpus, and does not improve the system's performance. We then resorted to automatic clustering based on the aspect model (Hofmann, 1999; Rooth et al., 1999). The training corpus used in clustering is the English portion of the selected FBIS corpus. Though automatically obtained verb clusters lead to further improvement in BLEU score, the total improvement from the semantic role features is not statistically significant. Because BLEU-4 is biased towards the adequacy of the MT outputs and may not effectively evaluate their fluency, it is desirable to give a more accurate evaluation of the sentence's fluency, which is the property that semantic role features are supposed to improve. To do this, we manually compare the outputs of the two log-linear models with and without the semantic role features. Our evaluation focuses on the completeness and ordering of the semantic roles, and *better*, *equal*, *worse* are tagged for each pair of MT outputs indicating the impact of the semantic role features. Table 2 shows the manual evaluation results based on the entire test set, and the improvement from SRF is significant at $p < 0.005$ based on a t-test. To illustrate how SRF impacts the translation results, Figure 8 gives 3 examples of the MT outputs with and without the SRFs.

5 Conclusion

This paper proposes two types of semantic role features for a Tree-to-String transducer: one models the reordering of the source-side semantic role sequence, and the other penalizes the deletion of a source-side semantic role. These semantic features

Source	Launching$_1$ New$_2$ Diplomatic$_3$ Offensive$_4$
SRF On	实施$_1$ 新的$_2$ 外交$_3$ 攻势$_4$
SRF Off	新的$_2$ 外交$_3$ 攻势$_4$

Source	It$_1$ is$_2$ therefore$_3$ necessary$_4$ to$_5$ speed$_6$ up$_7$ the$_8$ transformation$_9$ of$_{10}$ traditional$_{11}$ industries$_{12}$ with$_{13}$ high$_{14}$ technologies$_{15}$
SRF On	所以$_{123}$ 要$_4$ 加快$_{6,7}$ 高新技术$_{14,15}$ 改造$_9$ 传统产业$_{11,12}$
SRF Off	所以$_{123}$ 要$_4$ 高技术$_{14,15}$ ，加快$_{6,7}$ 传统产业$_{11,12}$ 改造$_9$

Source	A$_1$ gratifying$_2$ change$_3$ also$_4$ occurred$_5$ in$_6$ the$_7$ structure$_8$ of$_9$ ethnic$_{10}$ minority$_{11}$ cadres$_{12}$
SRF On	少数民族$_{10,11}$ 结构$_8$ 也$_4$ 发生$_5$ 可喜的$_2$ 变化$_3$
SRF Off	一个$_1$ 可喜的$_2$ 变化$_3$, 还在$_4$ 少数民族$_{10,11}$ 干部的 结构$_8$

Figure 8: Examples of the MT outputs with and without SRFs. The first and second example shows that SRFs improve the completeness and the ordering of the MT outputs respectively, the third example shows that SRFs improve both properties. The subscripts of each Chinese phrase show their aligned words in English.

and the Tree-to-String templates, trained based on a conditional log-linear model, are shown to significantly improve a basic TTS transducer's performance in terms of BLEU-4. To avoid BLEU's bias towards the adequacy of the MT outputs, manual evaluation is conducted for sentence fluency and significant improvement is shown by using the semantic role features in the log-linear model. Considering our semantic features are the most basic ones, using more sophisticated features (e.g., the head words and their translations of the source-side semantic roles) provides a possible direction for further experimentation.

Acknowledgments This work was funded by NSF IIS-0546554 and IIS-0910611.

References

Blunsom, Phil, Trevor Cohn, and Miles Osborne. 2008. A discriminative latent variable model for statistical machine translation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL-08)*, Columbus, Ohio.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL-01*, pages 132–139.

Dang, Hoa Trang, Karin Kipper, Martha Palmer, and

Joseph Rosenzweig. 1998. Investigating regular sense extensions based on intersective Levin classes. In *COLING/ACL-98*, pages 293–299, Montreal. ACL.

Dempster, A. P., N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the *EM* algorithm. *Journal of the Royal Statistical Society*, 39(1):1–21.

Fossum, Victoria, Kevin Knight, and Steven Abney. 2008. Using syntax to improveword alignment precision for syntax-based machine translation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, Columbus, Ohio. ACL.

Fung, Pascale, Zhaojun Wu, Yongsheng Yang, and Dekai Wu. 2006. Learning of Chinese/English semantic structure mapping. In *IEEE/ACL 2006 Workshop on Spoken Language Technology*, Aruba.

Galley, Michel, Mark Hopkins, Kevin Knight, and Daniel Marcu. 2004. What's in a translation rule? In *Proceedings of NAACL-04*, pages 273–280.

Galley, Michel, 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.

Graehl, Jonathan and Kevin Knight. 2004. Training tree transducers. In *Proceedings of NAACL-04*.

Hofmann, Thomas. 1999. Probabilistic latent semantic analysis. In *Uncertainity in Artificial Intelligence, UAI'99*, Stockholm.

Huang, Liang, 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.

Johnson, Christopher R., 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/.

Koehn, Philipp, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of NAACL-03*, Edmonton, Alberta.

Koehn, Philipp, 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, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL, Demonstration Session*, pages 177–180.

Koehn, Philipp. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP*, pages 388–395, Barcelona, Spain, July.

Liu, Ding and Daniel Gildea. 2008. Improved tree-to-string transducers for machine translation. In *ACL Workshop on Statistical Machine Translation (ACL08-SMT)*, pages 62–69, Columbus, Ohio.

Liu, Yang, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of COLING/ACL-06*, Sydney, Australia, July.

May, Jonathan and Kevin Knight. 2007. Syntactic re-alignment models for machine translation. In *Proceedings of EMNLP*.

Palmer, Martha, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL-02*.

Pradhan, Sameer, Wayne Ward, Kadri Hacioglu, James Martin, and Dan Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proceedings of NAACL-04*.

Rooth, Mats, Stefan Riezler, Detlef Prescher, Glenn Carroll, and Franz Beil. 1999. Inducing a semantically annotated lexicon via EM-based clustering. In *Proceedings of the 37th Annual Meeting of the ACL*, pages 104–111, College Park, Maryland.

Stolcke, Andreas. 2002. SRILM - an extensible language modeling toolkit. In *International Conference on Spoken Language Processing*, volume 2, pages 901–904.

Toutanova, Kristina, Aria Haghighi, and Christopher Manning. 2005. Joint learning improves semantic role labeling. In *Proceedings of ACL-05*, pages 589–596.

Wu, Dekai and Pascale Fung. 2009. Semantic roles for smt: A hybrid two-pass model. In *Proceedings of the HLT-NAACL 2009: Short Papers*, Boulder, Colorado.

Xue, Nianwen and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP*.

Zhu, Ciyou, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. 1994. L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization. Technical report, ACM Trans. Math. Software.