Tree-based Translation without Using Parse Trees

Feifei ZHAI, Jiajun ZHANG, Yu ZHOU and Chengqing ZONG
National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
{ffzhai, jjzhang, yzhou, cqzong}@nlpr.ia.ac.cn

ABSTRACT

Parse trees are indispensable to the existing tree-based translation models. However, there exist two major challenges in utilizing parse trees: 1) For most language pairs, it is hard to get parse trees due to the lack of syntactic resources for training. 2) Numerous parse trees are not compatible with word alignment which is generally learned by GIZA++. Therefore, a number of useful translation rules are often excluded. To overcome these two problems, in this paper we make a great effort to bypass the parse trees and induce effective unsupervised trees for tree-based translation models. Our unsupervised trees depend only on the word alignment without utilizing any syntactic resource or linguistic parser. Hence, they are very beneficial for the translation between resource-poor languages. Our experimental results have shown that the string-to-tree translation system using our unsupervised trees significantly outperforms the string-to-tree system using parse trees.

KEYWORDS : Tree-based translation; Unsupervised tree; EM algorithm.

Proceedings of COLING 2012: Technical Papers, pages 3037–3054,
COLING 2012, Mumbai, December 2012.

1 Introduction

Recently, ***tree-based models***[1] have been widely studied in statistical machine translation (SMT). The existing tree-based models include ***string-to-tree models*** (Galley et al., 2006; Marcu et al., 2006; Shen et al., 2008), ***tree-to-string models*** (Quirk et al., 2005; Liu et al., 2006; Huang et al., 2006;), and ***tree-to-tree models*** (Eisner, 2003; Ding and Palmer, 2005; Cowan et al., 2006; Zhang et al., 2008; Liu et al., 2009). Due to the effective use of syntactic information, tree-based models have achieved comparable (Liu et al., 2009) and even better performance over phrase-based models (Marcu et al., 2006).

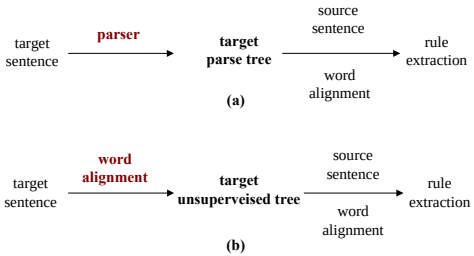

FIGURE 1 – Rule extraction for string-to-tree translation model: (a) using parse trees versus (b) using our unsupervised trees.

In the existing tree-based translation models, parse trees are essential to extracting translation rules. FIGURE 1(a) illustrates the rule extraction process of string-to-tree translation model. The parse tree is usually generated by a linguistic parser which is trained on a manually annotated corpus, such as Treebank. However, the manually annotated corpus is always too inadequate to fully display the strengths of tree-based models. In particular, traditional tree-based systems can not work at all for language pairs without any syntactic resource, which has greatly restricted their application.

From FIGURE 1(a), we can also discover that syntactic parsing is completely independent of word alignment. The separation of parser and alignment leads to a severe incompatibility problem between them. Together with the widely existing parsing errors, numerous useful translation rules are excluded during rule extraction.

To overcome the above two problems of current tree-based models, in this paper, we give up using parse trees and induce better alternatives for tree-based translation models. The alternative tree structures depend only on the word alignment without utilizing any syntactic resource or linguistic parser (see FIGURE 1(b) for illustration). Specifically, the entire process of inducing such tree structures for tree-based translation models is summarized as follows:

1. Based on a word aligned parallel corpus, we first transform those aligned bilingual sentence pairs into packed forests.

[1] The translation models using parse trees on one side or both sides are defined as tree-based models here.

2. Based on the obtained packed forests, we design an EM algorithm to learn an effective synchronous tree substitution grammar (STSG) and then acquire Viterbi tree structures according to the achieved STSG.

In step 1, in order to create a packed forest for a bilingual sentence pair, we first segment the sentence pair into several shorter ones to reduce the huge generation space of tree structures. Then, according to *frontier node assumption*, we compress all the tree structures with the largest number of frontier nodes into a packed forest. We will detail the process of constructing packed forest in Section 3. After all packed forests are generated, we exploit an EM algorithm in step 2 to learn an STSG and then generate Viterbi tree structures for translation. The adopted EM algorithm will be elaborated in Section 4.

Obviously, the above process of inducing tree structures is unsupervised. The syntactic resources and linguistic parsers are not necessary. Hence, comparing with parse trees, the proposed unsupervised trees can be applied to build translation models for more language pairs. Furthermore, by maximizing the number of frontier nodes, the unsupervised trees are compatible with word alignment and thus could achieve a better rule coverage for translation.

Since the existing tree-based translation models are usually restricted by parse trees, using unsupervised trees would be a promising direction for these models. To our best knowledge, this paper is the first effort to introduce effective unsupervised tree structures for tree-based translation models. The most significant contribution of this paper lies in this point. In order to achieve this goal, a series of useful techniques are employed innovatively and meaningfully in the paper. Moreover, the experimental results show that our unsupervised trees significantly outperform the parse trees in the state-of-the-art string-to-tree translation system.

2 Related Work

Our work focuses on inducing effective unsupervised tree structures, and meanwhile, resolving the incompatibility problem between tree structures and word alignment for tree-based translation.

Several researchers have studied unsupervised tree structure induction for different objectives. Blunsom et al. (2008, 2009, 2010) utilized Bayesian methods to learn synchronous context free grammar (SCFG) from a parallel corpus. The obtained SCFG grammar is further used in a phrase-based and hierarchical phrase-based system (Chiang, 2007). Denero and Uszkoreit (2011) adopted a parallel parsing model to induce unlabeled tree structures for syntactic pre-reordering. Different from above works, we concentrate on producing effective and labeled unsupervised trees for tree-based translation models. Moreover, since most of the current tree-based translation models are based on synchronous tree substitution grammar (STSG), our unsupervised trees are thus learned according to STSG, rather than SCFG.

On relieving the incompatibility problem between tree structures and word alignment for translation, previous works mainly focus on two directions:

One direction is to adapt the parse tree structure. Wang et al., (2007) binarized the parse trees and adopted an EM algorithm to select the best binary tree from their parallel binarization forest. Mi et al., (2008b) and Liu et al., (2009) compressed thousands of parse trees into packed forests. Zhang et al. (2011a) applied a CKY binarization method on parse trees to get binary forests for forest-to-string model. Burkett and Klein (2012) adopted a transformation-based method to learn a sequence of monolingual tree transformations for translation. They differ from our work in that

they were all based on parse trees. Compared with them, we construct effective unsupervised tree structures according to the word alignment and do not need any syntactic resource.

The other direction is to integrate the alignment information into parsing. Burkett and Klein (2008) and Burkett et al. (2010) made efforts to do joint parsing and alignment. They utilized the bilingual Treebank to train a joint model and achieved better results on both parsing and word alignment. Liu et al. (2012) re-trained the linguistic parsers bilingually based on word alignment. Our work is different from theirs in that we are pursuing better unsupervised tree structures for better translation performance.

As a whole, compared with previous works, our unsupervised trees are generated fully depending on word alignment. Therefore, by using our tree structures, the incompatibility problem between tree structures and word alignment can be well resolved.

3 Packed Forest Generation

In this section, we introduce how to compress all the reasonable tree structures into a packed forest for the given flat sentence. Packed forest is a compact representation of many tree structures. Generally, it is a pair $<V, E>$ where V is the set of *forest nodes* and E is the set of *hyperedges*. Each hyperedge $e \in E$ is a pair $< h(e), t(e) >$ where *h(e)* is its head node and *t(e)* denotes the vector of its tail nodes. FIGURE 2 illustrates a packed forest that encodes two different tree structures.

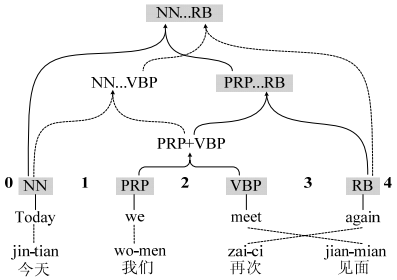

FIGURE 2 – An example of packed forest that encodes two different tree structures. In the FIGURE, shaded nodes denote frontier nodes

3.1 Space Reduction

Basically, there are an exponential number of possible tree structures for a given sentence. These tree structures result in a very huge packed forest. For example, considering a sentence of length L, there will be $0.5L(L+1)$ non-leaf nodes in the forest (each span corresponds to one node). In addition, a forest node covering ***m (m>=2)*** words emits ***m***-1 binary hyperedges[2], leading to $O(L^3)$ hyperedges in total[3]. Consequently, there would be an exponential number of parameters

[2] In our forest, the binary structure serves as the basic unit, which will be demonstrated later.

[3] There are a total of $L + \sum_{i=2}^{L}(L-i+1)\times(i-1) = \frac{1}{6}(L^3+5L)$ edges, including the edges linking to the leaf nodes.

for the STSG achieved from the forests. Such many parameters would cause a difficult estimation problem, especially for the EM algorithm adopted in this paper. Therefore, to reduce the huge generation space of tree structures, we first segment the bilingual sentence pair into several shorter ones and then impose *frontier node assumption* to construct the packed forest.

3.1.1 Bilingual Sentence Segmentation

Bilingual sentence segmentation is to segment a sentence pair into several short sub-sentence pairs whose source sub-sentence and target sub-sentence are translations to each other. Theoretically, in a sub-sentence pair, all included words cannot align to words outside it. However, since many words are wrongly aligned via the automatic word alignment, numerous correct aligned sub-sentence pairs are often excluded under this restriction. Therefore, to relax the restriction, we adopt the following constrains after analyzing the erroneous alignments[4]: (a) the bidirectional length ratios of a sub-sentence pair must be all smaller than 1:3; (b) as a sub-sentence pair, it must contain more than 4 words on each side; (c) in a sub-sentence pair, more than 30% words on each side must be aligned to its counterparts; (d) considering all the alignment links emitted by a sub-sentence, the erroneous ones (align to words outside the sub-sentence pair) account for at most 30% of all links.

To guarantee the segmentation accuracy, we only extract split point candidates based on the punctuations[5] which always denote the boundary of sub-sentences. Complying with the above constraints, we traverse all the split point candidates to search for the optimal split point with minimum number of wrongly aligned words (i.e., minimizing the number of words that align to words outside the sub-sentence pair). Then we segment the sentence pair into two shorter ones at that split point. We recursively segment the newly acquired sub-sentence pairs until no split point candidate is left. FIGURE 3(a) shows the segmentation result of an example sentence pair.

After bilingual sentence segmentation, only the spans inside the sub-sentence pairs are used in the forest. Under this condition, a large amount of useless spans are discarded and the forest is effectively simplified. For example, in FIGURE 3(b), the span “meet again, but” in the English sentence is discarded because it does not belong to any sub-sentence pair.

Note that the method of bilingual sentence segmentation we use here is only a simple segmentation strategy. It can also be substituted by any other segmentation methods. Additionally, after sentence segmentation, we realign words based on the sub-sentence pairs to get a new alignment where all words in a sub-sentence pair align to the words inside it.

3.1.2 Frontier Node Assumption

Bilingual sentence segmentation leads to a great space reduction for constructing packed forests. However, even after sentence segmentation, the generation space of tree structures would be still very large, especially when the sub-sentence is very long. In order to further simplify the space, we take advantage of the following assumption during forest construction:

Frontier Node Assumption*: The more frontier nodes the tree structure has, the more reasonable it is for translation.*

[4] The heuristic values in the constraints are chosen by a series of survey and experiments on a well-aligned corpus.
[5] We use {。 ， ： ； ？ ！} and {. , : ; ? !} as split anchors for Chinese and English, respectively. We take the position before and after the punctuations as split point candidates.

Frontier nodes are utilized to factor a tree structure into several fragments for rule extraction (Galley et al., 2004). Formally, a frontier node is a node that meets the following constraint: the span of the node and its dominated span at the other side are consistent with word alignment[6]. For example, in FIGURE 2, node *PRP…RB*'s span is {we meet again} and it dominates span {*我们 再次 见面*} at the other side. These two spans are consistent with word alignment. Therefore, node *PRP…RB* is a frontier node.

Our *frontier node assumption* makes sense in tree-based translation model. This is because with the purpose of achieving better rule coverage, we tend to extract small minimal rules as many as possible and generate larger rules by composing them. Maximizing the number of frontier nodes supports this goal, while producing many interior (non-frontier) nodes hinders it (DeNero and Klein, 2007). Hence, in the forest constructor, we follow this assumption and only consider the tree structures with the largest number of frontier nodes.

Denero and Uszkoreit (2011) utilized a similar heuristic to construct their unlabeled trees. They required that all spans in their trees must align continuously to the other side. Unlike their heuristic, our *frontier node assumption* only maximizes the number of frontier nodes. The interior nodes are also permitted in the tree structure, which is more flexible and appropriate for constructing forests.

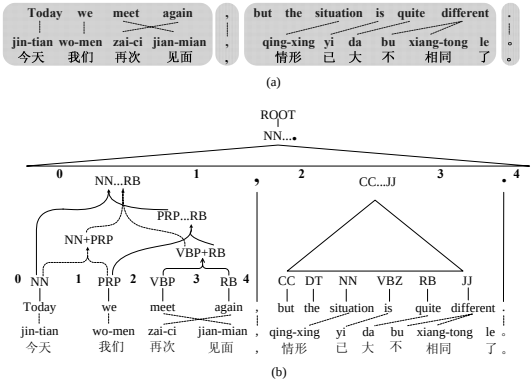

FIGURE 3 – (a) An example of bilingual sentence segmentation. (b) The ultimate packed forest of the example sentence pair in (a).

3.2 Node Labeling

To create packed forests for sentences, a problem that must be resolved is how to label the forest nodes without any syntactic knowledge. Xiong et al. (2006) showed that the boundary word of a phrase is a very effective indicator for phrase reordering. Zollmann and Vogel (2011) labeled hierarchical rules with word classes of boundary words and achieved better translation

[6] A node's dominated span at the other side refers to the minimum continuous span covering all the words that are reachable from the node via word alignment. Two spans are consistent with word alignment means that words in one span only align to words in the other span via word alignment, and vice versa.

performance. Inspired by their work, we combine word classes of boundary words to label forest nodes. We divide the non-leaf forest nodes into three groups: one-word node, dominating only one word in the sentence, and accordingly, two-word node, and multi-word node. Naturally, a one-word node is labeled by the class of its dominating word; a two-word node is labeled by combining the classes of the two words, such as "*C*1+*C*2"; a multi-word node, whose leftmost word's class is *C*1 and rightmost word's class is *C*n, is labeled with "*C*1…*C*n". In this paper, POS tags are employed to serve as the word classes[7]. For example, in FIGURE 3(b), the forest node covering phrase "we meet again" is a multi-word node and is labeled with "*PRP…RB*". Our labeling strategy is similar to (Zollmann and Vogel 2011). The difference is that we are labeling our forest nodes, while they labeled hierarchical rules to substitute the original single non-terminal *X*.

3.3 Forest Constructor

In tree-based translation models, the binary structure has shown its efficiency on improving translation quality (Wang et al., 2007; Zhang et al., 2011a). Inspired by this, we take the binary structure as the basic unit of our forest.

After sentence segmentation, we first build a forest for each sub-sentence pair. Initially, we create a POS node for each word and then perform a CKY-style algorithm to construct forests. FIGURE 4 illustrates the main process of building forest for a sub-sentence pair in FIGURE 3(a). From FIGURE 4 we can see that, the algorithm continuously inspects each span[8] in a bottom-up manner and creates forest nodes to represent the spans.

During the above process, we check every split point in each span and generate an edge[9] for that split point. To comply with the frontier node assumption, we only preserve the edges maximizing $F[i, j]$:

$$F[i, j] = \arg\max_k \{F[i,k] + F[k, j] + Fron[i, j]\} \quad (1)$$

where $F[i, j]$ denotes the number of frontier nodes in the sub-tree whose root node is span $[i, j]$. $k \in (i, j)$ refers to the split point of span $[i, j]$. $Fron[i, j]$ is an indicator function whose value is 1 if the node for span $[i, j]$ is a frontier node and 0 otherwise. Obviously, Equation (1) guarantees that the sub-tree rooted at span $[i, j]$ carries the largest number of fronter nodes. Consequently, in a bottom-up manner, when we arrive at the node covering the whole sentence, we can achieve all the tree structures with the larget number of frontier nodes.

For example, in FIGURE 4(c), span [0,3] (length L=3) has two split points and thus can be composed of span [0,1] and span [1,3], or span [0,2] and span [2,3]. However, just as FIGURE 4(c) shows, there are only 3 frontier nodes in the former case ($F[0,1] + F[1,3] + Fron[0,3] = 3$, here $Fron[0,3] = 0$ because node *NN…VBP[0,3]* is not a frontier node), while there are 4 frontier

[7] Practically, we need a supervised POS tagger, which impairs the unsupervised property of our tree structure to some extent. Actually, the POS tags can be substituted by any unsupervised word classes here. In future, we also plan to design an efficient algorithm to learn the node label automatically, rather than using a heuristic like here.

[8] Here the "span" is based on the POS nodes. For example in FIGURE 4, span 0-2 refers to the span of node sequence "NN PRP".

[9] As each split point corresponds to two adjacent smaller spans, we generate an edge to link these two spans. Therefore, each edge we create here contains a head node and only two tail nodes.

nodes in the latter one ($F[0,2]+F[2,3]+Fron[0,3]=4$). Therefore, we only preserve the edge maximizing $F[0,3]$ for node *NN…VBP[0,3]*, i.e., the edge composed of span [0,2] and span [2,3].

In most cases, a forest node could emit more than one edge with the same largest number of frontier nodes. For example, in FIGURE 4(d), node *NN…RB [0,4]* (length *L*=4) emits two edges with 7 frontier nodes. One links node *NN[0,1]* and *PRP…RB[1,4]*. The other one connects node *NN+PRP[0,2]* and *VPB+RB[2,4]*. We preserve both of these two edges for node *NN…RB [0,4]*. Finally, we achieve the packed forest for the example sub-sentence pair in FIGURE 4(d). In addition, during the forest construction process, the lower nodes and edges not chosen to create upper level nodes will be discarded, such as node *NN…VBP[0,3]* in FIGURE 4(d).

After we create a forest for each sub-sentence pair (named as sub-forest), we combine these sub-forests together to generate a final binary forest for the whole sentence pair. The combination is also performed by a similar CKY-style algorithm. The only difference is that the span in this CKY algorithm is based on the root node of sub-forests. Then we add a goal root node labeled "*ROOT*" to the forest which will be used in the EM algorithm. As an example, FIGURE 3(b) illustrates the final packed forest of the example sentence pair in FIGURE 3(a).

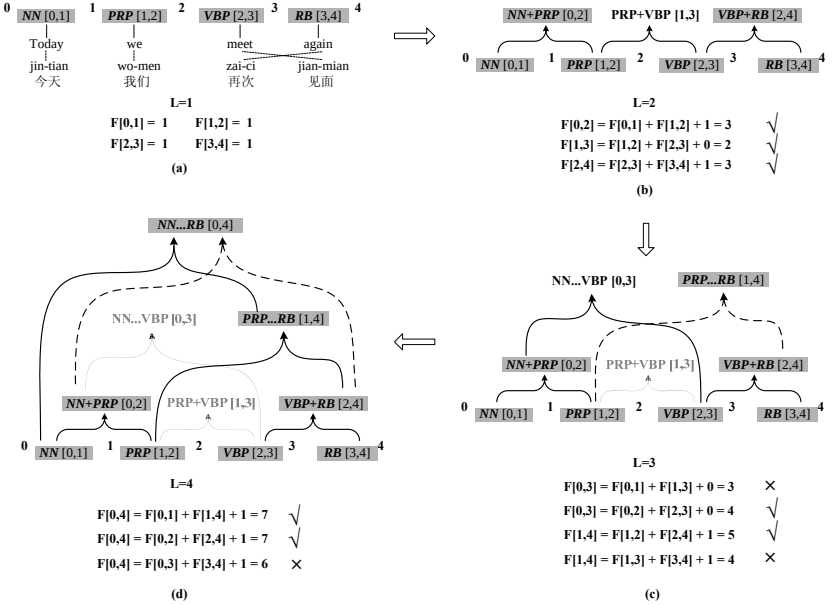

FIGURE 4 – The main process of building forest with the CKY-style algorithm. Here, shaded nodes denote frontier nodes. *L* refers to the length of span. F[i,j] denotes the number of frontier nodes in the sub-tree whose root node is span [i,j]. For example, the sub-tree that roots at span [1,4] in (c) contains 5 frontier nodes. This is because F[1,2] = 1 for node *PRP[1,2]*, and F[2,4] = 3 for node *VBP+RB[2,4]*, and node *PRP…RB [1,4]* is also a frontier node.

4 Learning Viterbi Tree via EM Algorithm

In this section, we design an EM algorithm to learn an effective synchronous tree substitution grammar (STSG) and then acquire Viterbi tree structures based on the STSG. Denero and Uszkoreit (2011) mentioned that their unlabeled tree structures can also be obtained by a similar method. However, their method is based on SCFG. Our work is different from theirs in that the EM algorithm is based on STSG. We use STSG here because most of the current tree-based translation models are based on STSG.

Given a parallel corpus with *n* sentence pairs, and the corresponding packed forest for each target sentence ***e***, we aim to search for a series of trees $(t_{e1}, t_{e2}, ..., t_{en})^*$ that maximize the likelihood of the whole corpus (t_e, f, a) [10], which is formulated as follows:

$$(t_{e1}, t_{e2}, ..., t_{en})^* = \arg\max_{(t_{e1}, t_{e2}, ..., t_{en})} \prod_{i=1}^{n} p(t_{ei}, f_i, a_i)$$

The probability of triple (t_{ei}, f_i, a_i) is further computed by aggregating the rule probabilities $p(r)$ in each derivation *d* in the set of all derivations *D*. That is

$$p(t_{ei}, f_i, a_i) = \sum_{D} \prod_{r \in d} p(r)$$

To get the derivation set *D*, we employ the algorithm of Mi et al., (2008a) to transform our induced packed forests into synchronous derivation forests. Practically, in order to reduce the complexity of the derivation forest, we only utilize the minimal STSG translation rules extracted by the method of Galley et al., (2004) and Mi et al., (2008b) to construct derivation forests[11].

Using the synchronous derivation forests, the rule probabilities are estimated by the inside-outside algorithm (Graehl and Knight, 2004). Here, leaf(***r***) and root(***r***) denote the leaf non-terminals and root node of rule ***r*** respectively. The inside and outside probabilities of forest node ***N*** are defined as follows,

$$p_{IN}(N) = \sum_{r \in R(N)} \left[p(r) \cdot \prod_{N_l \in leaf(r)} p_{IN}(N_l) \right]$$

$$p_{OUT}(N) = \sum_{r:\, N \in leaf(r)} \left[p(r) \cdot p_{OUT}(root(r)) \cdot \prod_{N_l \in leaf(r) - \{N\}} p_{IN}(N_l) \right]$$

where ***R(N)*** denotes the set of matched rules rooted at node ***N***. Therefore, the process of EM algorithm is shown as follows:

[10] In the triple, *te* refers to the target tree structures, *f* denotes the source language sentences, and *a* is the word alignment between them.

[11] We follow the highest attachment strategy in (Galley et al., 2006) to deal with unaligned words.

E-step: the expected count for each occurrence of rule ***r*** in a derivation forest is computed as:

$$p(r) \bullet p_{OUT}(root(r)) \bullet \prod_{N_l \in leaf(r)} p_{IN}(N_l)$$

M-step: the expected counts of rules, *c(r)*, are used to update the probabilities of rules:

$$p(r) = \frac{c(r)}{\sum_{r_a : root(r_a) = root(r)} c(r_a)}$$

After the EM algorithm, we traverse the derivation forest to obtain the Viterbi derivation d^* and its corresponding best tree structure. Then the acquired tree structures can be applied to any traditional tree-based translation system.

5 Experiments

5.1 Experimental Setup

In order to verify the effectiveness of our unsupervised tree structures, we compare them with linguistic parse trees based on string-to-tree translation. Here we experiment on Chinese-to-English translation, for which English parse trees can be easily obtained. Our training data is the FBIS corpus containing about 7.1 million Chinese words and 9.2 million English words. We generate the final symmetric word alignment using GIZA++ and the *grow-diag-final-and* balance strategy. We train a 5-gram language model on the target part of the training corpus and the Xinhua portion of English Gigaword corpus. We use the NIST MT 2003 evaluation data as the development set, and adopt NIST MT04 and MT05 as the test set. The final translation quality is evaluated in terms of case-insensitive BLEU-4 with shortest length penalty. The statistical significance test is performed using the re-sampling approach (Koehn, 2004).

Our baseline system is an in-house string-to-tree system (named ***s2t***) based on Galley et al. (2006) and Marcu et al. (2006). The English side of the training corpus is parsed with Berkeley parser (Petrov et al., 2006). We extract the minimal GHKM rules (Galley et al., 2004) and the rules of SPMT Model 1 (Marcu et al., 2006) with phrases up to length $L=5$ on the source side. Then we extract the composed rules by composing two or three adjacent minimal GHKM rules (Galley et al., 2006). The beam size of the decoder is set as 500. We further implement head binarization on the English parse trees and apply the achieved binary trees to another string-to-tree system (abbreviated as ***s2t-hb***) with the same settings of ***s2t***. In addition, we also run the state-of-the-art hierarchical phrase-based system Joshua (Li et al., 2009) for comparison.

For inducing our unsupervised tree structures, we use Urheen[12] to get the POS tags of the English corpus. Just as we described in section 3.1.1, we reuse GIZA++ and the *grow-diag-final-and* strategy to re-align words based on the sub-sentence pairs and then combine the alignment result together to get a new word alignment for the whole sentence pair. We perform the EM algorithm to capture the final tree structures by 20 iterations. Then we build a string-to-tree system using our induced unsupervised tree structures (abbreviated as ***s2t-IT***). Different from the above

[12] http://www.openpr.org.cn/index.php/NLP-Toolkit-for-Natural-Language-Processing/

baseline system, the beam size of our ***s2t-IT*** system is set as 300 to get a comparable translation speed to the baseline system.

Besides, using the new alignment, we also run the ***s2t*** and ***s2t-hb*** system with the same settings as the abovementioned ***s2t*** and ***s2t-hb*** systems (We mark the systems using the new word alignment as ***re-align*** systems).

5.2 Experimental Results

The translation results of different systems are shown in TABLE 1. As we can see, the performance of the baseline string-to-tree system significantly outperforms the hierarchical phrase-based system *Joshua*, which verifies the superiority of our baseline *s2t* system.

System		MT04	MT05	All
Joshua		30.71	27.86	29.59
s2t (baseline)		33.73*	30.25*	32.75*
s2t-hb		34.09	30.99*	32.92
re-align	***s2t***	33.53	29.30	32.29
	s2t-hb	33.88	30.49*	32.61
	s2t-IT	**34.71#**	**31.55#**	**33.53#**
Number of sentences		1788	1082	2870

TABLE 1 – Results (in case-insensitive BLEU-4 scores) of different systems. The "*" and "#" denote that the result are significantly better than the adjacent above system and all the other systems respectively ($p<0.01$).

TABLE 1 also demonstrates the effectiveness of binary structures. It can be clearly seen that whether we do re-alignment or not, the head binarization approach can always help to improve the *s2t* system (lines 2-5). Besides, as we can see from TABLE 1, the performances of the *re-align s2t* and *s2t-hb* system are slightly worse than the *s2t* and *s2t-hb* system. It indicates that the sentence segmentation method might be harmful to the traditional translation system. We will explore the reason in the next section.

The system using our induced unsupervised trees (*s2t-IT*) achieves the best performance among all the systems. It significantly outperforms the baseline *s2t* system by 0.98 and 1.3 BLEU points on MT04 and MT05 respectively. Furthermore, as shown in TABLE 1, even using the head binarization approach, the performance of the best *s2t-hb* system is still lower than that of our *s2t-IT* system by 0.61 BLEU points on the combined test set. Obviously, the above comparisons strongly demonstrate that our induced unsupervised trees are much more appropriate than parse trees for the string-to-tree translation model.

5.3 Analysis and Discussion

The improvement of translation performance has strongly verified the effectiveness of our induced unsupervised tree structures. We further conduct a series of deep analysis on the result.

We first adopt FIGURE 5, which depicts an example of our unsupervised tree structure and a traditional parse tree structure, to explain the superiority of our unsupervised trees. Comparing these two structures, we can see that our unsupervised tree structure carries more frontier nodes and thus can be factored into more small sub-structures. Consequently, the resulted minimal rules tend to be more general and smaller. For example, in FIGURE 5, rule (c) and (d) are the minimal

rules extracted from the unsupervised tree structure and the parse tree structure respectively to translate Chinese phrase "*有利于* (you-li-yu)". Obviously, rule (c) is much smaller and can be utilized without any limit while rule (d) cannot, since rule (d) requires that the translation after "*is conducive to*" must be dominated by an "***S***" node. Additionally, we can further acquire many big rules by composing several small minimal rules. On the above basis, our induced unsupervised trees are much more conducive to extracting both general enough rules and specific enough rules, which leads to a better rule coverage and translation quality.

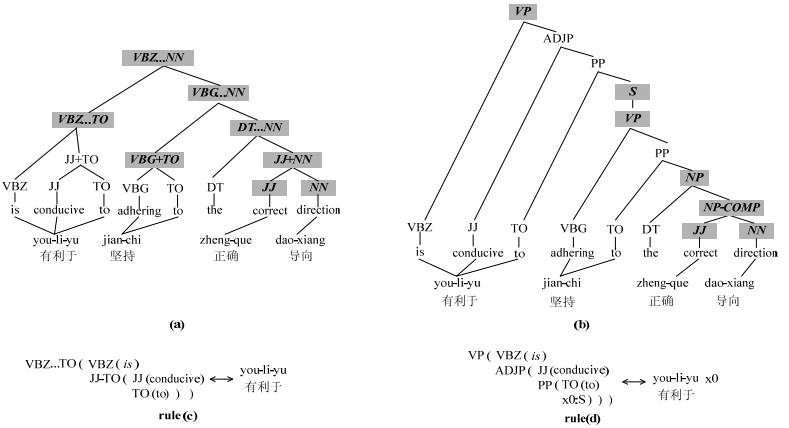

FIGURE 5 – Different tree structures and extracted example rules: (a) the unsupervised tree structures (b) the binary tree structures produced by berkeley parser (The node "NP-COMP" is created by the head binarization approach). The shaded nodes in the FIGURE denote frontier nodes. (c) and (d) are the minimal rules extracted from the structures in (a) and (b) respectively.

To further demonstrate the above analysis, TABLE 2 shows the average number of frontier nodes per tree structure (#Aver-Frontier-Nodes) and the grammar size (#RULES) of different systems. As we can see, the string-to-tree systems using parse trees benefit from the head binarization approach which helps to recall frontier nodes (from 33.9 to 40.4 for *s2t* system and 32.0 to 38.6 for the re-aligned *s2t* system).

System		#Aver-Frontier-Nodes	#RULES
s2t (baseline)		33.9	15.5M
s2t-hb		40.4	28.1M
re-align	***s2t***	32.0	13.8M
	s2t-hb	38.6	26.2M
	s2t-IT	**47.4**	**51.9M**

TABLE 2 – Average number of frontier nodes and grammar size for different string-to-tree systems.

Furthermore, using our induced unsupervised trees, it accounts for 47.4 frontier nodes on average while there are only 33.9 frontier nodes at most in the traditional linguistic parse trees. Obviously,

this comparison indicates that our unsupervised trees are more compatible with the word alignment and are beneficial to extracting more useful translation rules. Just as column "#RULES" shows, our *s2t-IT* system obtains a total of 51.9M rules while the baseline *s2t* system only gets 15.5M rules at most.

We have found that the sentence segmentation method might do harm to the traditional translation system in TABLE 1. TABLE 2 gives a faithful explanation on this phenomenon. As indicated by TABLE 2, after we do re-alignment, the number of frontier nodes decreases (from 33.9 to 32.0 for *s2t*) and the grammar size is reduced at the same time. We believe that the reduced grammar leads to the worse performance of the *re-align s2t* and *s2t-hb* system. Intuitively, the deterioration caused by sentence segmentation would also affect our *s2t-IT* system. However, our *s2t-IT* system still significantly outperforms the baseline *s2t* system. More work would be devoted to alleviate the influence of sentence segmentation.

We further investigate the used tags of tree nodes in our unsupervised trees. According to the statistics, there are a total of 2,862 tags for the non-leaf nodes in the final corpus of our unsupervised trees. With such many tags, a natural question is that does the grammar extracted from these tree structures suffer from a data sparseness problem? TABLE 3 answers this question in detail. In the TABLE, for example, line 2 denotes that the most frequent 143 tags (5% of all tags) account for 76.5% of all frontier nodes and 82.4% of all tree nodes. As illustrated in TABLE 3, 87.0% frontier nodes and 90.3% tree nodes are labeled with the most frequent 286 tags (10% of all tags), indicating that the vast majority of our translation rules are composed of these tags. Compared with the 70 tags[13] used in the linguistic parse trees, we believe our employed tags are both specific enough for distinguishing different rules and general enough for avoiding the data sparseness problem.

#tag num	#percentage of frontier nodes	#percentage of tree nodes
85(3%)	68.0%	75.8%
143(5%)	76.5%	82.4%
228(8%)	83.6%	87.7%
286(10%)	**87.0%**	**90.3%**
429(15%)	92.3%	94.2%
572(20%)	95.2%	96.4%
…	…	…

TABLE 3 – The proportion of frequently appearing tags in our induced tree structures.

Conclusion and Perspectives

In this paper, we propose effective unsupervised trees to substitute parse trees for tree-based translation models. Since current tree-based translation models are all driven by parse trees, this work creates a brand-new direction for them. We first roughly group the words into several sub-sentence pairs by a bilingual sentence segmentation method. After that, we compress all the reasonable tree structures of sentence pairs into packed forests under *frontier node assumption*. Finally, we design an EM algorithm to learn an effective STSG and then select a best tree structure for each sentence pair.

The unsupervised tree structures are constructed depending on the word alignment. Therefore, they are naturally compatible with word alignment and lead to a better rule coverage.

[13] There are 44 POS tags, 5 clausal tags and 21 phrasal tags for labeling the linguistic parse trees.

Experiments on string-to-tree translation system show that our unsupervised trees significantly outperform the parse trees. We believe that our method is quite beneficial for the translation between resource-poor languages.

In the future, we plan to conduct more experiments on other tree-based models, such as tree-to-string model and tree-to-tree model. Furthermore, we also plan to develop unsupervised methods to jointly induce the tree structure and word alignment for tree-based translation models. This issue is more difficult since the search space is much larger and we plan to employ Bayesian methods with sampling approach to fulfil this task.

Acknowledgments

The research work has been funded by the Natural Science Foundation of China under Grant No. 6097 5053 and the Hi-Tech Research and Development Program ("863" Program) of China under Grant No. 2011AA01A207. This research is also supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office. We also thank the anonymous reviewers for their valuable suggestions.

References

Blunsom, P. and Cohn, T. (2010). Inducing synchronous grammars with slice sampling. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*, pages 238–241, Los Angeles, California.

Blunsom, P., Cohn, T., Dyer, C., and Osborne, M. (2009). A gibbs sampler for phrasal synchronous grammar induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP)*, pages 782–790, Suntec, Singapore.

Blunsom, P., Cohn, T., and Osborne, M. (2008). Bayesian synchronous grammar induction. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems (NIPS) 21*, pages 161–168.

Burkett, D., Blitzer, J., and Klein, D. (2010). Joint parsing and alignment with weakly synchronized grammars. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 127–135, Stroudsburg, PA, USA. Association for Computational Linguistics.

Burkett, D. and Klein, D. (2008). Two languages are better than one (for syntactic parsing). In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 877–886, Stroudsburg, PA, USA. Association for Computational Linguistics.

Burkett, D. and Klein, D. (2012). Transforming trees to improve syntactic convergence. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 863–872, Jeju Island, Korea. Association for Computational Linguistics.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In

Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05, pages 263–270, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chiang, D. (2007). Hierarchical phrase-based translation. *Computational Linguistics*, 33 (2).

Chiang, D., Knight, K., and Wang, W. (2009). 11,001 new features for statistical machine translation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 218–226, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cohn, T. and Blunsom, P. (2009). A Bayesian model of syntax-directed tree to string grammar induction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 352–361, Singapore. Association for Computational Linguistics.

Cowan, B., Ku˘cerová, I., and Collins, M. (2006). A discriminative model for tree-to-tree translation. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 232–241, Sydney, Australia. Association for Computational Linguistics.

DeNeefe, S., Knight, K., Wang, W., and Marcu, D. (2007). What can syntax-based MT learn from phrase-based MT? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 755–763, Prague, Czech Republic. Association for Computational Linguistics.

DeNero, J. and Klein, D. (2007). Tailoring word alignments to syntactic machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 17–24, Prague, Czech Republic. Association for Computational Linguistics.

DeNero, J. and Uszkoreit, J. (2011). Inducing sentence structure from parallel corpora for reordering. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Ding, Y. and Palmer, M. (2005). Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 541–548, Ann Arbor, Michigan. Association for Computational Linguistics.

Eisner, J. (2003). Learning non-isomorphic tree mappings for machine translation. In *The Companion Volume to the Proceedings of 41st Annual Meeting of the Association for Computational Linguistics*, pages 205–208, Sapporo, Japan. Association for Computational Linguistics.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., and Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 961–968, Sydney, Australia. Association for Computational Linguistics.

Galley, M., Hopkins, M., Knight, K., and Marcu, D. (2004). What's in a translation rule? In Susan Dumais, D. M. and Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 273–280, Boston, Massachusetts, USA. Association for Computational Linguistics.

Graehl, J. and Knight, K. (2004). Training tree transducers. In Susan Dumais, D. M. and

Roukos, S., editors, *HLT-NAACL 2004: Main Proceedings*, pages 105–112, Boston, Massachusetts, USA. Association for Computational Linguistics.

Huang, L. and Chiang, D. (2007). Forest rescoring: Faster decoding with integrated language models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic. Association for Computational Linguistics.

Huang, L., Knight, K., and Joshi, A. (2006). A syntax-directed translator with extended domain of locality. In *Proceedings of the Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, CHSLP '06, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical Phrase-Based Translation, In *Proc. of HLT/NAACL 2003*.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In Lin, D. and Wu, D., editors, *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Li, Z., Callison-Burch, C., Dyery, C., Ganitkevitch, J., Khudanpur, S., Schwartz, L., Thornton, W. N. G., Weese, J., and Zaidan, O. F. (2009). Demonstration of joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the ACL-IJCNLP 2009 Software Demonstrations*, pages 25–28, Suntec, Singapore. Association for Computational Linguistics.

Liu, S., Li, C.-H., Li, M., and Zhou, M. (2012). Re-training monolingual parser bilingually for syntactic smt. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 854–862, Jeju Island, Korea. Association for Computational Linguistics.

Liu, Y., Liu, Q., and Lin, S. (2006). Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 609–616, Sydney, Australia. Association for Computational Linguistics.

Liu, Y., Lü, Y., and Liu, Q. (2009). Improving tree-to-tree translation with packed forests. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 558–566, Suntec, Singapore. Association for Computational Linguistics.

Marcu, D., Wang, W., Echihabi, A., and Knight, K. (2006). Spmt: Statistical machine translation with syntactified target language phrases. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 44–52, Sydney, Australia. Association for Computational Linguistics.

Mi, H. and Huang, L. (2008). Forest-based translation rule extraction. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 206–214, Honolulu, Hawaii. Association for Computational Linguistics.

Mi, H., Huang, L., and Liu, Q. (2008). Forest-based translation. In *Proceedings of ACL-08: HLT*, pages 192–199, Columbus, Ohio. Association for Computational Linguistics.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan. Association for Computational Linguistics.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Quirk, C., Menezes, A., and Cherry, C. (2005). Dependency treelet translation: Syntactically informed phrasal SMT. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 271–279, Ann Arbor, Michigan. Association for Computational Linguistics.

Shen, L., Xu, J., and Weischedel, R. (2008). A new string-to-dependency machine translation algorithm with a target dependency language model. In *Proceedings of ACL-08: HLT*, pages 577–585, Columbus, Ohio. Association for Computational Linguistics.

Snyder, B., Naseem, T., and Barzilay, R. (2009). Unsupervised multilingual grammar induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 73–81, Suntec, Singapore. Association for Computational Linguistics.

Wang, W., Knight, K., and Marcu, D. (2007). Binarizing syntax trees to improve syntax-based machine translation accuracy. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 746–754, Prague, Czech Republic. Association for Computational Linguistics.

Wang, W., May J., Knight, K., and Marcu, D. (2010). Re-structuring, re-labeling, and re-aligning for syntax-based machine translation. *Computational Linguistics*, 2010.

Xiong, D., Liu, Q., and Lin, S. (2006). Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney, Australia. Association for Computational Linguistics.

Xiong, D., Zhang, M., and Li, H. (2010). Learning translation boundaries for phrase-based decoding. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 136–144, Los

Angeles, California. Association for Computational Linguistics.

Zhai, F., Zhang, J., Zhou, Y., and Zong, C. (2011). Simple but Effective Approaches to Improving Tree-to-Tree Model. *MT-Summit-11*. pages 261-268.

Zhang, H., Fang, L., Xu, P., and Wu, X. (2011a). Binarized forest to string translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 835–845, Portland, Oregon, USA. Association for Computational Linguistics.

Zhang, H., Huang, L., Gildea, D., and Knight, K. (2006). Synchronous binarization for machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 256–263, New York City, USA. Association for Computational Linguistics.

Zhang, H., Zhang, M., Li, H., Aw, A., and Tan, C. L. (2009). Forest-based tree sequence to string translation model. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 172–180, Suntec, Singapore. Association for Computational Linguistics.

Zhang, J., Zhai, F., and Zong, C. (2011b). Augmenting string-to-tree translation models with fuzzy use of source-side syntax. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 204–215, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Zhang, J. and Zong, C. (2009). A framework for effectively integrating hard and soft syntactic rules into phrase based translation. In *Proceedings of the 23rd Pacific Asia Conference on Language, Information and Computation*, pages 579–588, Hong Kong. City University of Hong Kong.

Zhang, M., Jiang, H., Aw, A., Li, H., Tan, C. L., and Li, S. (2007). A Tree-to-Tree Alignment-based model for statistical Machine translation. *MT-Summit-07*. pages 535-542.

Zhang, M., Jiang, H., Aw, A., Li, H., Tan, C. L., and Li, S. (2008). A tree sequence alignment-based tree-to-tree translation model. In *Proceedings of ACL-08: HLT*, pages 559–567, Columbus, Ohio. Association for Computational Linguistics.

Zollmann, A. and Venugopal, A. (2006). Syntax augmented machine translation via chart parsing. In *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '06, pages 138–141, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zollmann, A. and Vogel, S. (2011). A word-class approach to labeling pscfg rules for machine translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1–11, Portland, Oregon, USA. Association for Computational Linguistics.