

Discriminative Word Alignment with Syntactic Features

Wei Chen

Supervisor: Dr. Stephan Vogel

Machine Translation Lab, Spring 2008

Abstract

This report introduces a study on syntactic features used in a discriminative word alignment model. The features are implemented on a state-of-the-art discriminative word alignment system. The syntactic features are extracted from parse trees. Three types of syntactic features are experimented in this work: one global tree path feature and two first order tree features. Experimental results show that the syntactic features are helpful for improving the word alignment accuracy on Chinese-English parallel sentences.

1. Introduction

1.1 Word Alignment

Parallel data is an important resource for machine translation systems. It contains three kinds of information: sentences in a source language, their translation sentences in a target language, and the alignment information between the sentence pairs. Automatic word alignment tools provide an efficient way for generating parallel data. The input to a word alignment model is a pair of source-target sentences, and the output is the word-level mapping (alignment) between the two sentences.

Automatic word alignment is not an easy task. Figure 1 shows an example parallel sentence pair. In this example, the source language is Chinese, and the target language is English. From this example, we could see the two typical problems of word alignment. First, the alignment is usually not a one-to-one mapping. Rather, the alignment can be one-to-many, many-to-one (as is shown in Figure 1, “between” and “and” map to the same Chinese word “yu”¹), many-to-many or even unaligned. Second, corresponding words in the two languages are not always in the same order (we could see a big distortion in the alignment in Figure 1). This idiosyncrasy between the languages brings great challenge to word alignment models.

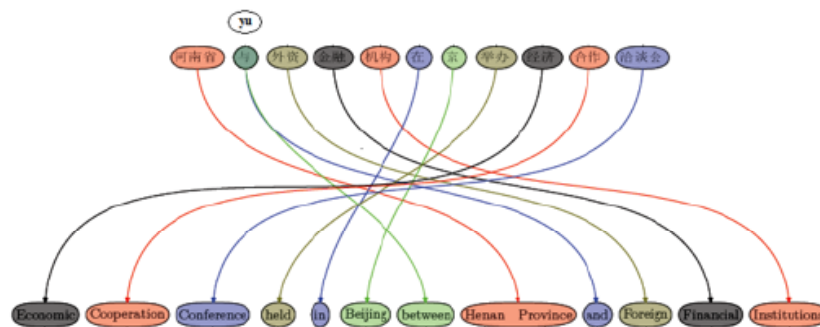


Figure 1. Parallel sentence Example. Adapted from (Ambati and Chen, 2007).

¹ We use the Pinyin for the transliteration of Chinese.

1.2 Word Alignment Models

Statistical word alignment models are usually trained on a large set of training data. There are generally two types of training models: generative models and discriminative models.

1.1.1 GIZA++ Alignment

GIZA++ is a widely used word alignment toolkit that implements a few existing word alignment methods (Brown et al., 1993; Vogel et al., 1996). This model formalizes the word alignment task as a noisy channel problem. It is a generative model which implements unsupervised training methods. Thus, GIZA++ training does not require hand-aligned data. Instead, a large amount of parallel data without alignment information is needed.

There are several problems in GIZA++ alignment. First, the alignment is not symmetric. That is, GIZA++ toolkit can only output one-directional alignment from a source language to a target language. And for each direction, the model only assumes many-to-one alignments, but not others. In order to get bi-directional alignment, we need to apply the training procedure once for each direction and use some combining algorithm to get a better bi-directional alignment. Existing combining operations on alignment matrices include *Union*, *Intersection*, *Sum*, and *Symmetrization* (Och, 2002), among many others. Second, since the model is generative, it is not flexible to incorporate new features.

1.1.2 Discriminative Alignment

In contrast to generative models, discriminative models are flexible for adding new features. But it requires hand-aligned data for supervised training. Several discriminative word alignment models have been proposed in recent years (Blunsom and Cohn 2006; Lacoste-Julien et al., 2006; Moore et al., 2006; Niehues and Vogel, 2008). Among these, Moore et al. (2006) use a log-linear model and a beam search. Lacoste-Julien et al. (2006) use larger fertilities and first order features, and they also used a posteriors of alignment model (Liang et al., 2006). Blunsom and Cohn (2006) use two Conditional Random Fields (CRFs) to model alignments from each direction, which give them an asymmetric alignment, so an additional combination of the two directions needs to be applied. Compared to (Blunsom and Cohn, 2006), Niehues and Vogel (2008) also use CRF, but their approach is symmetric. No heuristics for combining the two alignment directions are needed. There are four types of features used in their model: local features, fertility features, first order features, and phrase features (Niehues, 2007). Local feature set captures the features within a small observance window. It includes GIZA++ output as a feature, which gives much improvement. Fertility features capture the probability of a word being mapped to zero, one or multiple words on the other language side. First order features capture the dependencies between alignments in a neighborhood. Finally, phrase features capture the context dependencies of an alignment. There are also part of speech (POS) features which can be placed into both local features and first order features. Our syntactic features are implemented on top of the Niehues and Vogel (2008) model.

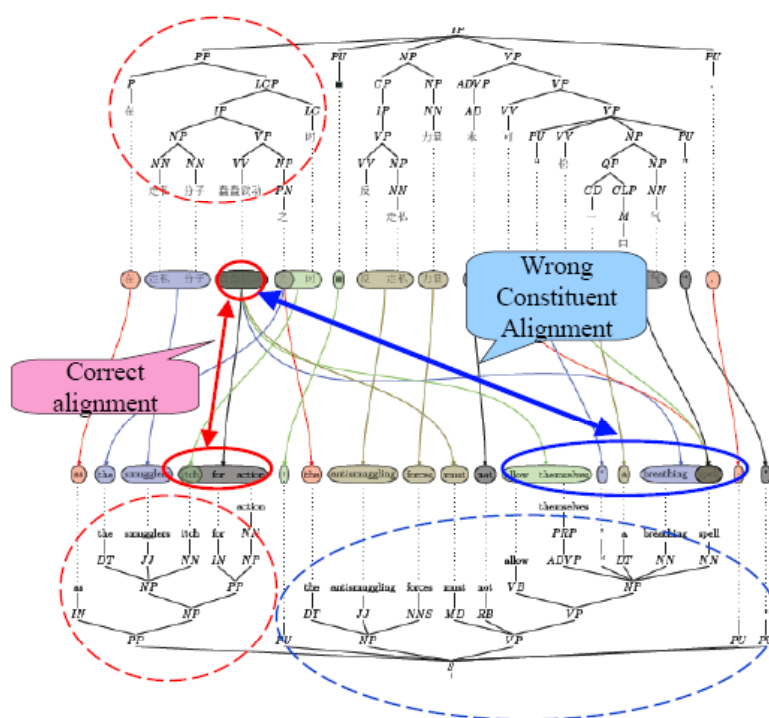
The goal of our work is to extend the local features and first order features to incorporate syntactical information for Chinese-English word alignment.

2 Some Problems of Chinese-English Alignment

Chinese and English have very different syntactical structures. We have seen from Figure 1 that there can be large distortions in the alignments. We believe this distortion comes from syntactical information, which is the basic motivation of this work. There are three major problems in current Chinese-English word alignment: (1) cross constituent alignment; (2) long-string-distance dependency; (3) un-captured similar syntactic structures. This section demonstrates examples of these problems.

2.1 Cross-Constituent Alignment

Cross-constituent alignment is almost the most commonly occurred error in Chinese-English word alignments. A constituent can be represented as a sub-tree in the phrase structure tree. As is shown in Figure 2, Both the Chinese parse tree and the English parse tree contain a sub-tree rooted at PP (marked by the dotted red circles). The words in these two sub-trees actually align to their counterparts in PP-sub-tree on the other language side. We could see that a word on the Chinese side (marked by a solid-line red circle) is linked to four English words. But only one of these is correct, which lies in the corresponding sub-tree on the English side.



Feature 2. Example of Cross-constituent alignment.

2.2 Long-String-Distance Dependency

Without syntactic information, word alignment largely depends on string-level

constraints. There are some long-distance dependencies that cannot be well modeled in traditional models. But with syntactic information, we could see that there is only long-distance on the string level, but actually short-distance on the syntactic tree level. Figure 3 shows an example of this kind. Two Chinese words “zai” and “xia” are highly correlated; they are commonly appeared in Chinese just as “from” and “to” in English. Also, these two words are usually translated to one single English word. However, it is not easy to capture their dependencies because these two words usually have a long string-level distance. In the example shown in Figure 3, these two words have a string distance of 7. But the dependency may be captured as a local feature from the parse tree. The tree distance (defined as the shortest path between the two corresponding leaf nodes) is only 3, where the closest possible tree distance between any two leaf nodes is 2. Hence, the long-distance dependency on the string level can be modeled as a local dependency on the parse tree level.

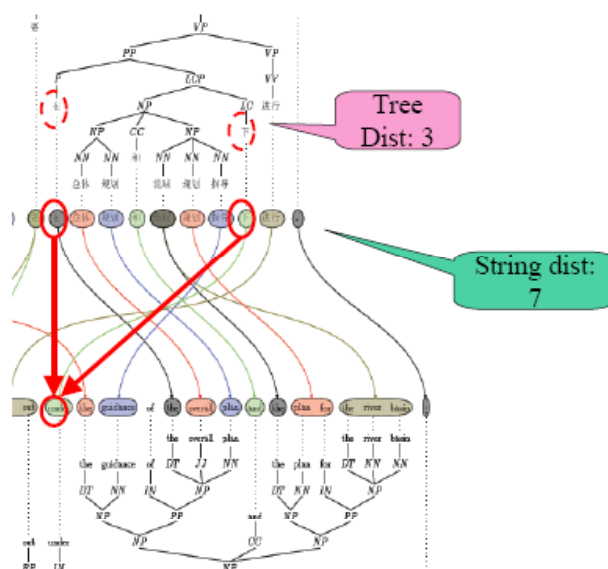


Figure 3. Long-string-distance but short-tree-distance word alignment.

2.3 Un-captured Similar Syntactic Structures

The third type of alignment error is similar to the first one, but may be harder to capture. Figure 4 shows an example. The dotted line circles marks two sub-trees that have similar structures, which indicates a constituent-level alignment. The two sub-trees are all rooted at PP. On the Chinese side, the PP is rewritten as P and NP, where on the English side, the PP is rewritten as IN and NP. It is intuitive to align the P with the IN, which actually gives a correct alignment. Other alignments, which are shown in the figure, are wrong. However, this requires us to define some similarity functions to capture these similar structures, which is still an open question.

3 Syntactic Features

From the observations in the previous section, we could expect that some syntactic feature might help in word alignment. We implement three kinds of syntactical

features. These features are extracted directly from parse trees. So the first step is to parse Chinese and English sentences using the Stanford parser. Then the parse tree will be cleaned and stored in a tree data structure. Finally, various kinds of parse tree features can be extracted.

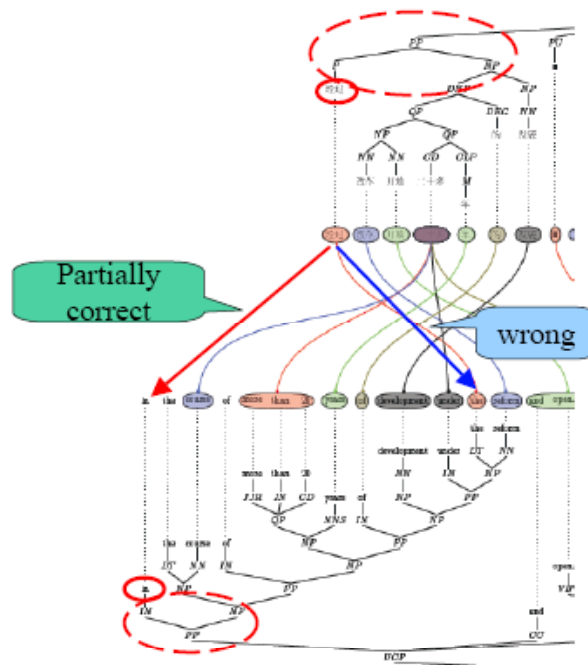


Figure 4. Un-captured similar syntactic structures

3.1 Tree Path Feature

Tree path feature is a global tree feature. It models the generation history of a word in the parse tree. For example, a Chinese word “tiaozheng” and its corresponding English translation “restructuring” has the following two generation paths, respectively:

Tiaozheng: (VP PP IP VP NP NP NP) => (VP PP IP VP NP)

Restructuring: (VP VP PP NNP NP) => (VP PP NP)

The right hand side of the arrows illustrates the compressed paths, where the repeated non-terminal nodes are skipped. By compression, we usually can get a pair of similar paths for correctly aligned words. The reason we perform path compression is that most English trees are deeper than Chinese trees, due to the differences between English and Chinese grammars. Therefore, English paths are usually much longer than Chinese paths. In order to get a robust similarity function for the Chinese-English paths, we perform the path compression to reduce the noise. Apart from the compression, we also do some other pre-processing to the paths based on some heuristics of the two languages. These include a set of operations: (1) get rid of all the SBAR’s, which do not exist in Chinese grammar; (2) get rid of all the CLP’s; (3) if we encounter a QP, we only look at symbols before the QP and neglect the others after the QP; (4) get rid of all the X’s; (5) if we encounter a POS, we only look at symbols before the POS and neglect the others after the POS.

The tree path feature function is written as:

$f(\{y_{ij}\}) = \text{sim}(\text{path}(e_i), \text{path}(f_j)) \cdot \delta(y_{ij})$, where e_i and f_j stands for the source word and target word, respectively; y_{ij} denotes a hidden node in the CRF graph which corresponds to an alignment in the alignment matrix; $\delta(\bullet)$ is the indicator function; The similarity function $\text{sim}()$ is the similarity function of the alignment path pairs. We currently only implemented the length similarity function which is defined as:

$$\delta(s - d \leq b \leq s + d), \text{ where } s = \min\{\text{length}(\text{path}(e_i)), \text{length}(\text{path}(f_j))\},$$

$$d = \max\{\text{length}(\text{path}(e_i)), \text{length}(\text{path}(f_j))\}, b = 1/s.$$

On our hand-aligned data, 81.9% of the alignments have this feature, so we expect this feature to be helpful. In the CRF graph, each factored node connects to only one hidden node. So this feature is computationally equivalent to the local features in (Niehues and Vogel 2008). There are in total $O(n^2)$ number of features, where n denotes the length of the sentences.

3.2 Tree First Order Feature-1

There are two types of tree first order features. The first type of the feature is only implemented on already connected nodes by the first order features in (Niehues and Vogel 2008). The original first order feature connects an alignment (e_i, f_j) with its neighbor (e_{i+s}, f_{j+t}) , where s and t are pre-defined directions. In our implementation, we look at 6 directions. The tree first order feature-1 captures whether the already connected nodes have some distance property in tree-distance measure or not. The feature function is written as:

$$f(\{y_{ij}, y_{(i+s)(j+t)}\}) = \delta(\text{dist}(e_i, e_{i+s}) - \text{dist}(f_j, f_{j+t}) = d) \cdot \delta(y_{ij}) \cdot \delta(y_{(i+s)(j+t)}) \quad , \quad \text{where}$$

$\{y_{ij}, y_{(i+s)(j+t)}\}$ represents a factored node connecting to two hidden nodes; $\text{dist}(\bullet, \bullet)$

returns the tree distance between two words; $d \in \{0, 1, 2, 3, \text{others}\}$. No additional

connections are needed in this feature implementation. The total number of features is

30 times the number of tree path features; it is still on the order of n^2 .

3.3 Tree First Order Feature-2

The second type of feature is similar to the original first order features, but we use tree directions rather than string directions. We still use the 6 directions in the original features. The feature function is defined as:

$$f(\{y_{ij}, y_{pq}\}) = \delta(y_{ij}) \cdot \delta(y_{pq}), \quad p \in \text{tree-add}(i, s), \quad q \in \text{tree-add}(j, t),$$

where $\text{tree-add}(\bullet, *)$ returns a set of indices that has a required tree distance s to the current word. To implement the $\text{tree-add}(\bullet, *)$ function, we use Dijkstra’s shortest path algorithm. The total number of features in this feature set depends on the size of the set returned by $\text{tree-add}(\bullet, *)$. The worst case can be reached when $\text{tree-add}(\bullet, *)$ returns the whole set of indices in the sentence, which implies a very flat tree. In this case, the number of tree features can be $O(n^4)$. The real number of features in our experiments can easily reach several thousand. In order to deal with this run time issue, we set an upper bound on the size of the set returned by $\text{tree-add}(\bullet, *)$, which would bring the upper bound of the number of features back to $O(n^2)$.

4 Experimental Results

We use 200 hand-aligned sentences for the discriminative training. But in order to have the GIZA++ features, we use a much larger amount of data for unsupervised training and use the parameters to extract GIZA++ features for the 200 sentences. Similar procedure also needs to be done during decoding time. Training is done in two steps: (1) maximum likelihood (2) alignment error rate (AER) optimization. The training and testing results are shown in Table 1. Our baselines are obtained by the recommended setting from (Niehues, 2007). The experiments are done without the POS features, because POS features usually cause the experiments to run very slowly. We finally training POS features with the scaling factors for other features fixed in a separate step. The experimental results are shown in Table 2.

	Precision	Recall	F-Score	Train (AER)	Test (AER)
Baseline	70.10	52.47	62.77	35.08	37.23
Path	72.34	56.03	63.15	34.57	36.85
FOF-1	75.13	54.84	63.39	34.24	36.61
FOF-2	71.36	57.10	63.44	33.40	36.56

Table 1. Feature performances (without POS features)

	Precision	Recall	F-Score	Train (AER)	Test (AER)
Baseline	69.84	64.73	67.19	22.47	32.81
Tree features	70.60	64.09	67.20	21.10	32.80

Table 2. Feature performances (with POS features)

5 Conclusions and Future Work

In this work, we extended an existing discriminative word alignment model to have syntactic features for Chinese-English parallel data. We mainly designed and implemented three kinds of tree features:

- (1) a global tree path feature which models the generation history of a word in the parse tree
- (2) a local indicator feature which models the tree-distance similarity for existing first order connections
- (3) a tree-based first order feature which models alignment dependencies within a neighborhood.

There is still plenty of space for further improvements:

- (1) Segmentation of Chinese numbers might help obtain better alignment. We tried segmentation, but no discriminative training experiment is done.
- (2) The similarity function in the tree path feature needs further research. We only implemented a length-similarity function, but a more comprehensive one which captures some mappings between the English and Chinese POS is needed.
- (3) We may need to modify the directions for Tree FOF-2 and capture a multi-layer feature. For example, we could have a feature that involves both a distance-3 neighborhood and a distance-5 neighborhood. But again, the run time may be a problem. Also, when there are too many features, we may overfit.
- (4) There might be a good similarity function for sub-trees, which may capture the “un-captured similar syntactic structure” problem.
- (5) The learning rates of the features need to be tuned. We currently apply the suggested learning rates in (Niehues, 2007), which might not be the optimal for syntactical features.

6 References

- Ambati, V. and W. Chen. 2007. *Cross Lingual Syntax Projection for Resource-Poor Languages*. <http://www.cs.cmu.edu/~weichen/survey.pdf>.
- Blunsom P. and Cohn T. 2006. Discriminative Word Alignment with Conditional Random Fields. In *ACL'06*. pp. 65-72. Sydney, Australia.
- Brown, P. F., S. Della Pietra, V. J. Della Pietra, R. L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*. 19(2):263-311.
- Lacoste-Julien, S., B. Taskar, D. Klein, M. I. Jordan. 2006. Word Alignment via Quadratic Assignment. In *HLT-NAACL'06*. New York, USA.
- Liang, P., B. Taskar, D. Klein. 2006. Alignment by Agreement. In *HLT-NAACL'06*. pp. 104-110. New York, USA.
- Moore, R. C., W. Yih, A. Bode. 2006. Improved Discriminative Bilingual Word Alignment. In *ACL'06*. pp. 513-520. Sydney, Australia.
- Niehues, J. 2007. Discriminative Word Alignment Models. Diplomarbeit at Universitat Karlsruhe (TH).
- Niehues, J. and S. Vogel. 2008. Discriminative Word Alignment via Alignment Matrix

Modeling. In *the Second Workshop on Syntax and Structure in Translation (SSST) at the Association for Computational Linguistics (ACL)*. 2008. Columbus, Ohio.

Och, F. J. 2002. *Statistical Machine Translation: From Single Word Models to Alignment Templates*. PhD thesis, Computer Science Department, RWTH Aachen, Germany.

Vogel, S, H. Ney, C. Tillmann. 1996. HMM-Based Word Alignment in Statistical Translation. In *COLING'96*. pp. 836-841. Copenhagen, Denmark.