Improving Word Alignment by Semi-supervised Ensemble

Shujian Huang[1], **Kangxi Li**[2], **Xinyu Dai**[1], **Jiajun Chen**[1]
[1]State Key Laboratory for Novel Software Technology at Nanjing University
Nanjing 210093, P.R.China
{huangsj,daixy,chenjj}@nlp.nju.edu.cn
[2]School of Foreign Studies, Nanjing University
Nanjing 210093, P.R.China
richardlkx@126.com

Abstract

Supervised learning has been recently used to improve the performance of word alignment. However, due to the limited amount of labeled data, the performance of "pure" supervised learning, which only used labeled data, is limited. As a result, many existing methods employ features learnt from a large amount of unlabeled data to assist the task. In this paper, we propose a semi-supervised ensemble method to better incorporate both labeled and unlabeled data during learning. Firstly, we employ an ensemble learning framework, which effectively uses alignment results from different unsupervised alignment models. We then propose to use a semi-supervised learning method, namely Tri-training, to train classifiers using both labeled and unlabeled data collaboratively and further improve the result. Experimental results show that our methods can substantially improve the quality of word alignment. The final translation quality of a phrase-based translation system is slightly improved, as well.

1 Introduction

Word alignment is the process of learning bilingual word correspondences. Conventional word alignment process is treated as an unsupervised learning task, which automatically learns the correspondences between bilingual words using an EM style algorithm (Brown et al., 1993; Vogel et al., 1996; Och and Ney, 2003). Recently, supervised learning methods have been used to improve the performance. They firstly re-formalize word alignment as some kind of classification task. Then the labeled data is used to train the classification model, which is finally used to classify unseen test data (Liu et al., 2005; Taskar et al., 2005; Moore, 2005; Cherry and Lin, 2006; Haghighi et al., 2009).

It is well understood that the performance of supervised learning relies heavily on the feature set. As more and more features are added into the model, more data is needed for training. However, due to the expensive cost of labeling, we usually cannot get as much labeled word alignment data as we want. This may limit the performance of supervised methods (Wu et al., 2006). One possible alternative is to use features learnt in some unsupervised manner to help the task. For example, Moore (2005) uses statistics like log-likelihood-ratio and conditional-likelihood-probability to measure word associations; Liu et al. (2005) and Taskar et al. (2005) use results from IBM Model 3 and Model 4, respectively.

Ayan and Dorr (2006) propose another way of incorporating unlabeled data. They first train some existing alignment models, e.g. IBM Model4 and Hidden Markov Model, using unlabeled data. The results of these models are then combined using a maximum entropy classifier, which is trained using labeled data. This method is highly efficient in training because it only makes decisions on alignment links from existing models and avoids searching the entire alignment space.

In this paper, we follow Ayan and Dorr (2006)'s idea of combining multiple alignment results. And we use more features, such as bi-lexical features, which help capture more information from unlabeled data. To further improve the decision making during combination, we propose to use a semi-supervised strategy, namely Tri-training (Zhou and Li, 2005), which ensembles three classifiers using both labeled and unlabeled data. More specifically, Tri-training iteratively trains three classifiers and labels all the unlabeled instances. It then uses some instances among the unlabeled ones to expand the labeled training set of each in-

Proceedings of the Fourteenth Conference on Computational Natural Language Learning, pages 135–143,
Uppsala, Sweden, 15-16 July 2010.

dividual classifier. As word alignment task usually faces a huge parallel corpus, which contains millions of unlabeled instances, we develop specific algorithms to adapt Tri-training for this large scale task.

The next section introduces the supervised alignment combination framework; Section 3 presents our semi-supervised learning algorithm. We show the experiments and results in Section 4; briefly overview related work in Section 5 and conclude in the last section.

2 Word Alignment as a Classification Task

2.1 Modeling

Given a sentence pair $(\mathbf{e}, \mathbf{f})$, where $\mathbf{e} = e_1, e_2, \ldots, e_I$ and $\mathbf{f} = f_1, f_2, \ldots, f_J$, an alignment link $a_{i,j}$ indicates the translation correspondence between words e_i and f_j. Word alignment is to learn the correct alignment A between $\mathbf{e}$ and $\mathbf{f}$, which is a set of such alignment links.

As the number of possible alignment links grows exponentially with the length of $\mathbf{e}$ and $\mathbf{f}$, we restrict the candidate set using results from several existing alignment models. Note that, all the models we employ are unsupervised models. We will refer to them as sub-models in the rest of this paper.

Let $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ be a set of alignment results from sub-models; A_I and A_U be the intersection and union of these results, respectively. We define our learning task as: for each alignment link $a_{i,j}$ in the candidate set $A_C = A_U - A_I$, deciding whether $a_{i,j}$ should be included in the alignment result. We use a random variable $y_{i,j}$ (or simply y) to indicate whether an alignment link $a_{i,j} \in A_C$ is correct. A Maximum Entropy model is employed to directly model the distribution of y. The probability of y is defined in Formula 1, where $h_m(y, \mathbf{e}, \mathbf{f}, \mathcal{A}, i, j)$ is the m^{th} feature function, and λ_m is the corresponding weight.

$$p(y|\mathbf{e}, \mathbf{f}, \mathcal{A}, i, j) = \frac{exp\sum_{m=1}^{M} \lambda_m h_m(y, \mathbf{e}, \mathbf{f}, \mathcal{A}, i, j)}{\sum_{\hat{y} \in \{0,1\}} exp\sum_{m=1}^{M} \lambda_m h_m(\hat{y}, \mathbf{e}, \mathbf{f}, \mathcal{A}, i, j)} \quad (1)$$

While Ayan and Dorr (Ayan and Dorr, 2006) make decisions on each alignment link in A_U, we take a different strategy by assuming that all the alignment links in A_I are correct, which means alignment links in A_I are always included in the combination result. One reason for using this strategy is that it makes no sense to exclude an alignment link, which all the sub-models vote for including. Also, links in A_I usually have a good quality (In our experiment, A_I can always achieve an accuracy higher than 96%). On the other hand, because A_I is decided before the supervised learning starts, it will be able to provide evidence for making decisions on candidate links.

Also note that, Formula 1 is based on the assumption that given A_I, the decision on each y is independent of each other. This is the crucial point that saves us from searching the whole alignment space. We take this assumption so that the Tri-training strategy can be easily applied.

2.2 Features

For ensemble, the most important features are the decisions of sub-models. We also use some other features, such as POS tags, neighborhood information, etc. Details of the features for a given link $a_{i,j}$ are listed below.

Decision of sub-models: Whether $a_{i,j}$ exists in the result of k^{th} sub-model A_k. Besides individual features for each model, we also include features describing the combination of sub-models' decisions. For example, if we have 3 sub-models, there will be 8 features indicating the decisions of all the sub-models as 000, 001, 010, ..., 111.

Part of speech tags: POS tags of previous, current and next words in both languages. We also include features describing the POS tag pairs of previous, current and next word pairs in the two languages.

Neighborhood: Whether each neighbor link exists in the intersection A_I. Neighbor links refer to links in a 3*3 window with (i, j) in the center.

Fertilities: The number of words that e_i (or f_j) is aligned to in A_I.

Relative distance: The relative distance between e_i and f_j, which is calculated as $abs(i/I - j/J)$.

Conditional Link Probability (CLP) : The conditional link probability (Moore, 2005) of e_i

and f_j. CLP of word e and f is estimated on an aligned parallel corpus using Formula 2,

$$CLP_d(e, f) = \frac{link(e, f) - d}{cooc(e, f)} \quad (2)$$

where $link(e, f)$ is the number of times e and f are linked in the aligned corpus; $cooc(e, f)$ is the number of times e and f appear in the same sentence pair; d is a discounting constant which is set to 0.4 following Moore (2005). We estimate these counts on our set of unlabeled data, with the union of all sub-model results A_U as the alignment. Union is used in order to get a better link coverage. Probabilities are computed only for those words that occur at least twice in the parallel corpus.

bi-lexical features: The lexical word pair e_i-f_j.

Lexical features have been proved to be useful in tasks such as parsing and name entity recognition. Taskar et al. (2005) also employ similar bi-lexical features of the top 5 non-punctuation words for word alignment. Using bi-lexicons for arbitrary word pairs will capture more evidence from the data; although it results in a huge feature set which may suffer from data sparseness. In the next section, we introduce a semi-supervised strategy will may alleviate this problem and further improve the learning procedure.

3 Semi-supervised methods

Semi-supervised methods aim at using unlabeled instances to assist the supervised learning. One of the prominent achievements in this area is the Co-training paradigm proposed by Blum and Mitchell (1998). Co-training applies when the features of an instance can be naturally divided into two sufficient and redundant subsets. Two weak classifiers can be trained using each subset of features and strengthened using unlabeled data. Blum and Mitchell (1998) prove the effectiveness of this algorithm, under the assumption that features in one set is conditionally independent of features in the other set. Intuitively speaking, if this conditional independence assumption holds, the most confident instance of one classifier will act as a random instance for the other classifier. Thus it can be safely used to expand the training set of the other classifier.

The standard Co-training algorithm requires a naturally splitting in the feature set, which is hard to meet in most scenarios, including the task of word alignment. Variations include using random split feature sets or two different classification algorithms. In this paper, we use the other Co-training style algorithm called Tri-training, which requires neither sufficient and redundant views nor different classification algorithms.

3.1 Tri-training

Similar with Co-training, the basic idea of Tri-training (Zhou and Li, 2005) is to iteratively expand the labeled training set for the next-round training based on the decisions of the current classifiers. However, Tri-training employs three classifiers instead of two. To get diverse initial classifiers, the training set of each classifier is initially generated via bootstrap sampling from the original labeled training set and updated separately. In each round, these three classifiers are used to classify all the unlabeled instances. An unlabeled instance is added to the training set of any classifier if the other two classifiers agree on the labeling of this example. So there is no need to explicitly measure the confidence of any individual classifier, which might be a problem for some learning algorithms. Zhou and Li (2005) also give a terminate criterion derived from PAC analysis. As the algorithm goes, the number of labeled instances increases, which may bring in more bi-lexical features and alleviate the problem of data sparseness.

3.2 Tri-training for Word Alignment

One crucial problem for word alignment is the huge amount of unlabeled instances. Typical parallel corpus for word alignment contains at least hundreds of thousands of sentence pairs, with each sentence pair containing tens of instances. That makes a large set of millions of instances. Therefore, we develop a modified version of Tri-training algorithm using sampling techniques, which can work well with such large scale data. A sketch of our algorithm is shown in Figure 1.

The algorithm takes original labeled instance set L, unlabeled sentence set S_U, sub-model results $\mathcal{A}_s$ for each s in S_U and a sampling ratio r as input. F_k represents the k^{th} classifier. Variables with superscript i represent their values during the i^{th} iteration.

Line 2 initializes candidate instance set $A_{C,s}$ of each sentence s to be the difference set between

Input: L, S_U, $\mathcal{A}_s$ for each s and sampling ratio r.

1: **for all** sentence s in S_U **do**
2: $A^0_{C,s} \leftarrow A_{U,s} - A_{I,s}$ //initializing candidate set
3: **end for**
4: **for all** $l \in \{1,2,3\}$ **do**
5: $L^0_l \leftarrow Subsample(L, 0.33)$
6: $F^0_l \leftarrow Train(L^0_l)$
7: **end for**
8: **repeat**
9: **for all** $l \in \{1,2,3\}$ **do**
10: Let $m, n \in \{1,2,3\}$ and $m \neq n \neq l$; $L^i_l = \emptyset$
11: **for all** sentence s in S_U **do**
12: **for all** instance a in $A^{i-1}_{C,s}$ **do**
13: **if** $F^{i-1}_m(a) = F^{i-1}_n(a)$ **then**
14: $A^{i-1}_{C,s} \leftarrow A^{i-1}_{C,s} - \{(a, F^{i-1}_m(a))\}$
15: $L^i_l \leftarrow L^i_l \cup \{(a, F^{i-1}_m(a))\}$
16: **end if**
17: **end for**
18: **end for**
19: **end for**
20: **for all** $l \in \{1,2,3\}$ **do**
21: $L^i_l \leftarrow Subsampling(L^i_l, r) \cup L^{i-1}_l$
22: $F^i_l \leftarrow Train(L^i_l)$
23: $A^i_{C,s} \leftarrow A^{i-1}_{C,s}$
24: **end for**
25: **until** all $A^i_{C,s}$ are unchanged or empty

Output: $F(x) \leftarrow arg\max_{y \in \{0,1\}} \sum_{l:F_l(x)=y} 1$

Figure 1: Modified Tri-training Algorithm

$A_{U,s}$ and $A_{I,s}$. In line 5-6, sub-samplings are performed on the original labeled set L and the initial classifier F^0_l is trained using the sampling results. In each iteration, the algorithm labels each instance in the candidate set $A^i_{C,s}$ for each classifier with the other two classifiers trained in last iteration. Instances are removed from the candidate set and added to the labeled training set (L^i_l) of classifier l, if they are given the same label by the other two classifiers (line 13-16).

A sub-sampling is performed before the labeled training set is used for training (line 21), which means all the instances in L^i_l are accepted as correct, but only part of them are added into the training set. The sampling rate is controlled by a parameter r, which we empirically set to 0.01 in all our experiments. The classifier is then re-trained using the augmented training set L^i_l (line 22). The algorithm iterates until all instances in the candidate sets get labeled or the candidate sets do not change since the last iteration (line 25). The resulting classifiers can be used to label new instances via majority voting.

Our algorithm differs from Zhou and Li (2005) in the following three aspects. First of all, comparing to the original bootstrap sampling initialization, we use a more aggressive strategy, which actually divides the original labeled set into three parts. This strategy ensures that initial classifiers are trained using different sets of instances and maximizes the diversity between classifiers. We will compare these two initializations in the experiments section. Secondly, we introduce sampling techniques for the huge number of unlabeled instances. Sampling is essential for maintaining a reasonable growing speed of training data and keeping the computation physically feasible. Thirdly, because the original terminate criterion requires an error estimation process in each iteration, we adapt the much simpler terminate criterion of standard Co-training into our algorithm, which iterates until all the unlabeled data are finally labeled or the candidate sets do not change since the last iteration. In other words, our algorithm inherits both the benefits of using three classifiers and the simplicity of using Co-training style termination criterion. Parallel computing techniques are also used during the processing of unlabeled data to speed up the computation.

Source	Usage	Sent. Pairs	Cand. Links
LDC	Train	288111	8.8M
NIST'02	Train	200	5,849
NIST'02	Eval	291	7,797

Table 1: Data used in the experiment

4 Experiments and Results

4.1 Data and Evaluation Methodology

All our experiments are conducted on the language pair of Chinese and English. For training alignment systems, a parallel corpus coming from LDC2005T10 and LDC2005T14 is used as unlabeled training data. Labeled data comes from NIST Open MT Eval'02, which has 491 labeled sentence pairs. The first 200 labeled sentence pairs are used as labeled training data and the rest are used for evaluation (Table 1). The number of candidate alignment links in each data set is also listed in Table 1. These candidate alignment links are generated using the three sub-models described in Section 4.2.

The quality of word alignment is evaluated in terms of alignment error rate (AER) (Och and Ney, 2003), classifier's accuracy and recall of correct decisions. Formula 3 shows the definition of AER, where P and S refer to the set of possible and sure alignment links, respectively. In our experiments,

ModelName	AER_Dev	AER_Test	Accuracy	Recall	F_1
Model4C2E	0.4269	0.4196	0.4898	0.3114	0.3808
Model4E2C	0.3715	0.3592	0.5642	0.5368	0.5502
BerkeleyAl.	0.3075	0.2939	0.7064	0.6377	0.6703
Model4GDF	0.3328	0.3336	0.6059	0.6184	0.6121
Supervised	**0.2291**	**0.2430**	**0.8124**	**0.7027**	**0.7536**

Table 2: Experiments of Sub-models

ModelName	AER_Dev	AER_Test	Accuracy	Recall	F_1
Supervised	0.2291	**0.2430**	**0.8124**	0.7027	0.7536
BerkeleyAl.	0.3075	0.2939	0.7064	0.6377	0.6703
Tri-Bootstrap^0	0.2301	0.2488	0.8030	0.6858	0.7398
Tri-Divide^0	0.2458	0.2525	0.8002	0.6630	0.7251
Tri-Bootstrap	**0.2264**	0.2468	0.7934	0.7449	0.7684
Tri-Divide	0.2416	0.2494	0.7832	**0.7605**	**0.7717**

Table 3: Experiments of Semi-supervised Models

we treat all alignment links as sure links.

$$AER = 1 - \frac{|A \cap P| + |A \cap S|}{|A| + |S|} \quad (3)$$

We also define a F_1 score to be the harmonic mean of classifier's accuracy and recall of correct decisions (Formula 4).

$$F_1 = \frac{2 * accuracy * recall}{accuracy + recall} \quad (4)$$

We also evaluate the machine translation quality using unlabeled data (in Table 1) and these alignment results as aligned training data. We use multi-references data sets from NIST Open MT Evaluation as development and test data. The English side of the parallel corpus is trained into a language model using SRILM (Stolcke, 2002). Moses (Koehn et al., 2003) is used for decoding. Translation quality is measured by BLEU4 score ignoring the case.

4.2 Experiments of Sub-models

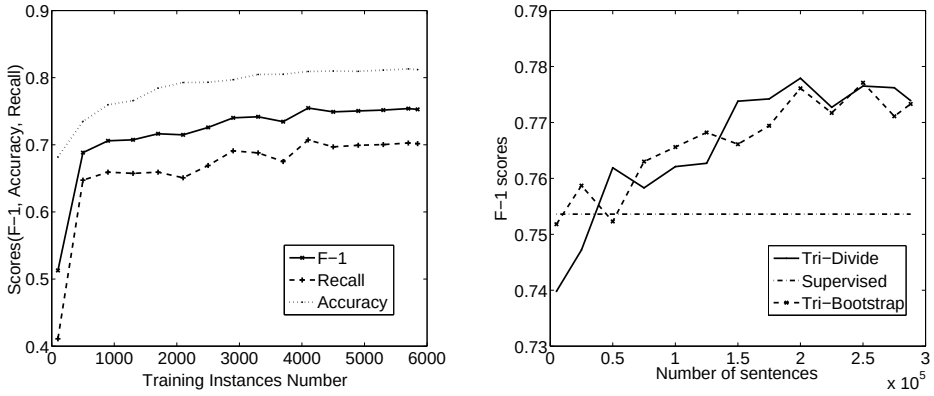
We use the following three sub-models: bidirectional results of Giza++ (Och and Ney, 2003) Model4, namely *Model4C2E* and *Model4E2C*, and the joint training result of BerkeleyAligner (Liang et al., 2006) (*BerkeleyAl.*). To evaluate AER, all three data sets listed in Table 1 are combined and used for the unsupervised training of each sub-model.

Table 2 presents the alignment quality of those sub-models, as well as a supervised ensemble of them, as described in Section 2.1. We use the symmetrized IBM Model4 results by the grow-diag-final-and heuristic as our baseline (*Model4GDF*). Scores in Table 2 show the great improvement of supervised learning, which reduce the alignment error rate significantly (more than 5% AER points from the best sub-model, i.e. BerkeleyAligner). This result is consistent with Ayan and Dorr (2006)'s experiments. It is quite reasonable that supervised model achieves a much higher classification accuracy of 0.8124 than any unsupervised sub-model. Besides, it also achieves the highest recall of correct alignment links (0.7027).

4.3 Experiments of Semi-supervised Models

We present our experiment results on semi-supervised models in Table 3. The two strategies of generating initial classifiers are compared. *Tri-Bootstrap* is the model using the original bootstrap sampling initialization; and *Tri-Divide* is the model using the dividing initialization as described in Section 3.2. Items with superscripts 0 indicate models before the first iteration, i.e. initial models. The scores of BerkeleyAligner and the supervised model are also included for comparison.

In general, all supervised and semi-supervised models achieve better results than the best sub-model, which proves the effectiveness of labeled training data. It is also reasonable that initial models are not as good as the supervised model, because they only use part of the labeled data for training. After the iterative training, both the two



(a) (b)

Figure 2: (a) Experiments on the Size of Labeled Training Data in Supervised Training; (b) Experiments on the Size of Unlabeled Data in Tri-training

Tri-training models get a significant increase in recall. We attribute this to the use of bi-lexical features described in Section 2.2. Analysis of the resulting model shows that the number of bi-lexical features increases from around 300 to nearly 7,800 after Tri-training. It demonstrates that semi-supervised algorithms are able to learn more bi-lexical features automatically from the unlabeled data, which may help recognize more translation equivalences. However, we also notice that the accuracy drops a little after Tri-training. This might also be caused by the large set of bi-lexical features, which may contain some noises.

In the comparison of initialization strategies, the dividing strategy achieves a much higher recall of 0.7605, which is also the highest among all models. It also achieves the best F_1 score of 0.7717, higher than the bootstrap sampling strategy (0.7684). This result confirms that diversity of initial classifiers is important for Co-training style algorithms.

4.4 Experiments on the Size of Data

4.4.1 Size of Labeled Data

We design this experiment to see how the size of labeled data affects the supervised training procedure. Our labeled training set contains 5,800 training instances. We randomly sample different sets of instances from the whole set and perform the supervised training.

The alignment results are plotted in Figure 2a. Basically, both accuracy and recall increase with the size of labeled data. However, we also find that the increase of all the scores gets slower when the number of training instances exceeds 3,000. One possible explanation for this is that the training set itself is too small and contains redundant instances, which may prevent further improvement. We can see in the Section 4.4.2 that the scores can be largely improved when more data is added.

4.4.2 Size of Unlabeled Data

For better understanding the effect of unlabeled data, we run the Tri-training algorithm on unlabeled corpus of different sizes. The original unlabeled corpus contains about 288 thousand sentence pairs. We create 12 sub-corpus of it with different sizes by selecting certain amounts of sentences from the beginning. Our smallest sub-corpus consists of the first 5,000 sentence pairs of the original corpus; while the largest sub-corpus contains the first 275 thousand sentence pairs. The alignment results on these different sub-corpus are evaluated (See Figure 2b).

The result shows that as the size of unlabeled data grows, the F_1 score of *Tri-Divide* increases from around 0.74 to 0.772. The F_1 score of *Tri-Bootstrap* also gets a similar increase. This proves that adding unlabeled data does help the learning process. The result also suggests that when the size of unlabeled data is small, both *Tri-Bootstrap* and *Tri-Divide* get lower scores than the supervised model. This is because the Tri-training models only use part of the labeled data for the training of each individual classifier, while the supervised model use the whole set. We can see that when there are more than 50 thousand unlabeled sentence pairs, both Tri-training models outperform the supervised model significantly.

ModelName	Dev04	Test05	Test06	Test08
Model4C2E	24.54	17.10	17.52	14.59
Model4E2C	26.54	19.00	20.18	16.56
BerkeleyAl.	26.19	20.08	19.65	16.70
Model4GDF	26.75	20.67	20.58	17.05
Supervised	**27.07**	20.00	19.47	16.13
Tri-Bootstrap	26.88	20.49	20.76	**17.31**
Tri-Divide	27.04	**20.96**	**20.79**	17.18

Table 4: Experiments on machine translation (BLEU4 scores in percentage)

Note that, both experiments on data size show some unsteadiness during the learning process. We attribute this mainly to the random sampling we use in the algorithm. As there are, in all, about 8.8 million instances , it is highly possible that some of these instances are redundant or noisy. And because our random sampling does not distinguish different instances, the quality of resulting model may get affected if these redundant or noisy instances are selected and added to the training set.

4.5 Experiments on Machine Translation

We compare the machine translation results of each sub-models, supervised models and semi-supervised models in Table 4. Among sub-models, BerkeleyAligner gets better BLEU4 scores in almost all the data sets except TEST06, which agrees with its highest F_1 score among all sub-models. The supervised method gets the highest BLEU score of 27.07 on the dev set. However, its performance on the test sets is a bit lower than that of BerkeleyAligner.

As we expect, our two semi-supervised models achieve highest scores on almost all the data sets, which are also higher than the commonly used grow-diag-final-and symmetrization of IBM Model 4. More specifically, *Tri-Divide* is the best of all systems. It gets a dev score of 27.04, which is comparable with the highest one (27.07). *Tri-Divide* also gets the highest BLEU scores on Test05 and Test06 (20.96 and 20.79, respectively), which are nearly 1 point higher than all sub-models. The other Tri-training model, *Tri-Bootstrap*, gets the highest score on Test08, which is also significantly better than those sub-models.

Despite the large improvement in F_1 score, our two Tri-training models only get slightly better score than the well-known *Model4GDF*. This kind of inconsistence between AER or F_1 scores and BLEU scores is a known issue in machine translation community (Fraser and Marcu, 2007). One possible explanation is that both AER or F_1 are 0-1 loss functions, which means missing one link and adding one redundant link will get the same penalty. And more importantly, every wrong link receives the same penalty under these metrics. However, these different errors may have different effects on the machine translation quality. Thus, improving alignment quality according to AER or F_1 may not directly lead to an increase of BLEU scores. The relationship among these metrics are still under investigation.

5 Related work

Previous work mainly focuses on supervised learning of word alignment. Liu et al. (2005) propose a log-linear model for the alignment between two sentences, in which different features can be used to describe the alignment quality. Moore (2005) proposes a similar framework, but with more features and a different search method. Other models such as SVM and CRF are also used (Taskar et al., 2005; Cherry and Lin, 2006; Haghighi et al., 2009). For alignment ensemble, Wu and Wang (2005) introduce a boosting approach, in which the labeled data is used to calculate the weight of each sub-model.

These researches all focus on the modeling of alignment structure and employ some strategy to search for the optimal alignment. Our main contribution here is the use Co-training style semi-supervised methods to assist the ensemble learning framework of Ayan and Dorr (2006). Although we use a maximum entropy model in our experiment, other models like SVM and CRF can also be incorporated into our learning framework.

In the area of semi-supervised learning of word alignment, Callison-Burch et al. (2004) compare the results of interpolating statistical machine

translation models learnt from labeled and unlabeled data, respectively. Wu et al. (2006) propose a modified boosting algorithm, where two different models are also trained using labeled and unlabeled data respectively and interpolated. Fraser and Marcu (2006) propose an EMD algorithm, where labeled data is used for discriminative reranking. It should be pointed out that these pieces of work all use two separate processes for learning with labeled and unlabeled data. They either train and interpolate two separate models or rerank previously learnt models with labeled data only. Our proposed semi-supervised strategy is able to incorporate both labeled and unlabeled data in the same process, which is in a different line of thinking.

6 Conclusions and Future Work

Semi-supervised techniques are useful when there is a large amount of unlabeled data. In this paper, we introduce a semi-supervised learning method, called Tri-training, to improve the word alignment combination task. Although experiments have proved the effectiveness of our methods, there is one defect that should be mentioned. As we previously assume that all the decisions on alignment links are independent of each other (in Section 2.1), our model are only able to capture link level evidence like bi-lexical features. Some global features, such as final word fertility, cannot be integrated into the current framework. In the future, we plan to apply our semi-supervised strategy in more complicated learning frameworks, which are able to capture those global features.

Currently we use a random sampling to handle the 8.8 million instances. We will also explore better and more aggressive sampling techniques, which may lead to more stable training results and also enable us to process larger corpus.

Acknowledgments

The authors would like to thank Dr. Ming Li, Mr. Junming Xu and the anonymous reviewers for their valuable comments. This work is supported by the National Fundamental Research Program of China(2010CB327903) and the Scientific Research Foundation of Graduate School of Nanjing University(2008CL08).

References

Necip Fazil Ayan and Bonnie J. Dorr. 2006. A maximum entropy approach to combining word alignments. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 96–103, Morristown, NJ, USA. Association for Computational Linguistics.

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pages 92–100. Morgan Kaufmann Publishers.

Peter F. Brown, Stephen Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The mathematic of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Chris Callison-Burch, David Talbot, and Miles Osborne. 2004. Statistical machine translation with word- and sentence-aligned parallel corpora. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 175, Morristown, NJ, USA. Association for Computational Linguistics.

Colin Cherry and Dekang Lin. 2006. Soft syntactic constraints for word alignment through discriminative training. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 105–112, Morristown, NJ, USA. Association for Computational Linguistics.

Alexander Fraser and Daniel Marcu. 2006. Semi-supervised training for statistical word alignment. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 769–776, Morristown, NJ, USA. Association for Computational Linguistics.

Alexander Fraser and Daniel Marcu. 2007. Measuring word alignment quality for statistical machine translation. *Comput. Linguist.*, 33(3):293–303.

Aria Haghighi, John Blitzer, and Dan Klein. 2009. Better word alignments with supervised itg models. In *Association for Computational Linguistics*, Singapore.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *HLT-NAACL*.

Percy Liang, Benjamin Taskar, and Dan Klein. 2006. Alignment by agreement. In Robert C. Moore, Jeff A. Bilmes, Jennifer Chu-Carroll, and Mark Sanderson, editors, *HLT-NAACL*. The Association for Computational Linguistics.

Yang Liu, Qun Liu, and Shouxun Lin. 2005. Log-linear models for word alignment. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 459–466, Morristown, NJ, USA. Association for Computational Linguistics.

Robert C. Moore. 2005. A discriminative framework for bilingual word alignment. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 81–88, Morristown, NJ, USA. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51.

A. Stolcke. 2002. Srilm - an extensible language modeling toolkit. In *Proceedings of International Conference on Spoken Language Processing*, page 901 904.

Ben Taskar, Simon Lacoste-Julien, and Dan Klein. 2005. A discriminative matching approach to word alignment. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 73–80, Morristown, NJ, USA. Association for Computational Linguistics.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 836–841.

Hua Wu and Haifeng Wang. 2005. Boosting statistical word alignment. In *Proceedings of MT SUMMIT X*, pages 364–371, Phuket Island, Thailand, September.

Hua Wu, Haifeng Wang, and Zhanyi Liu. 2006. Boosting statistical word alignment using labeled and unlabeled data. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 913–920, Sydney, Australia, July. Association for Computational Linguistics.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. volume 17, pages 1529–1541, Piscataway, NJ, USA. IEEE Educational Activities Department.