Boosting Statistical Word Alignment Using Labeled and Unlabeled Data

Hua Wu Haifeng Wang Zhanyi Liu
Toshiba (China) Research and Development Center
5/F., Tower W2, Oriental Plaza, No.1, East Chang An Ave., Dong Cheng District
Beijing, 100738, China
{wuhua, wanghaifeng, liuzhanyi}@rdc.toshiba.com.cn

Abstract

This paper proposes a semi-supervised boosting approach to improve statistical word alignment with limited labeled data and large amounts of unlabeled data. The proposed approach modifies the supervised boosting algorithm to a semi-supervised learning algorithm by incorporating the unlabeled data. In this algorithm, we build a word aligner by using both the labeled data and the unlabeled data. Then we build a pseudo reference set for the unlabeled data, and calculate the error rate of each word aligner using only the labeled data. Based on this semi-supervised boosting algorithm, we investigate two boosting methods for word alignment. In addition, we improve the word alignment results by combining the results of the two semi-supervised boosting methods. Experimental results on word alignment indicate that semi-supervised boosting achieves relative error reductions of 28.29% and 19.52% as compared with supervised boosting and unsupervised boosting, respectively.

1 Introduction

Word alignment was first proposed as an intermediate result of statistical machine translation (Brown et al., 1993). In recent years, many researchers build alignment links with bilingual corpora (Wu, 1997; Och and Ney, 2003; Cherry and Lin, 2003; Wu et al., 2005; Zhang and Gildea, 2005). These methods *unsupervisedly* train the alignment models with unlabeled data.

A question about word alignment is whether we can further improve the performances of the word aligners with available data and available alignment models. One possible solution is to use the boosting method (Freund and Schapire, 1996), which is one of the ensemble methods (Dietterich, 2000). The underlying idea of boosting is to combine simple "rules" to form an ensemble such that the performance of the single ensemble is improved. The AdaBoost (**Ada**ptive **Boost**ing) algorithm by Freund and Schapire (1996) was developed for supervised learning. When it is applied to word alignment, it should solve the problem of building a reference set for the unlabeled data. Wu and Wang (2005) developed an *unsupervised* AdaBoost algorithm by automatically building a pseudo reference set for the unlabeled data to improve alignment results.

In fact, large amounts of unlabeled data are available without difficulty, while labeled data is costly to obtain. However, labeled data is valuable to improve performance of learners. Consequently, *semi-supervised learning*, which combines both labeled and unlabeled data, has been applied to some NLP tasks such as word sense disambiguation (Yarowsky, 1995; Pham et al., 2005), classification (Blum and Mitchell, 1998; Thorsten, 1999), clustering (Basu et al., 2004), named entity classification (Collins and Singer, 1999), and parsing (Sarkar, 2001).

In this paper, we propose a *semi-supervised* boosting method to improve statistical word alignment with both limited labeled data and large amounts of unlabeled data. The proposed approach modifies the supervised AdaBoost algorithm to a semi-supervised learning algorithm by incorporating the unlabeled data. Therefore, it should address the following three problems. The first is to build a word alignment model with both labeled and unlabeled data. In this paper, with the labeled data, we build a *supervised model* by directly estimating the parameters in

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 913–920,
Sydney, July 2006. ©2006 Association for Computational Linguistics

the model instead of using the Expectation Maximization (EM) algorithm in Brown et al. (1993). With the unlabeled data, we build an *unsupervised model* by estimating the parameters with the EM algorithm. Based on these two word alignment models, an *interpolated model* is built through linear interpolation. This interpolated model is used as a learner in the semi-supervised AdaBoost algorithm. The second is to build a reference set for the unlabeled data. It is automatically built with a modified "refined" combination method as described in Och and Ney (2000). The third is to calculate the error rate on each round. Although we build a reference set for the unlabeled data, it still contains alignment errors. Thus, we use the reference set of the labeled data instead of that of the entire training data to calculate the error rate on each round.

With the interpolated model as a learner in the semi-supervised AdaBoost algorithm, we investigate two boosting methods in this paper to improve statistical word alignment. The first method uses the unlabeled data only in the interpolated model. During training, it only changes the distribution of the labeled data. The second method changes the distribution of both the labeled data and the unlabeled data during training. Experimental results show that both of these two methods improve the performance of statistical word alignment.

In addition, we combine the final results of the above two semi-supervised boosting methods. Experimental results indicate that this combination outperforms the unsupervised boosting method as described in Wu and Wang (2005), achieving a relative error rate reduction of 19.52%. And it also achieves a reduction of 28.29% as compared with the supervised boosting method that only uses the labeled data.

The remainder of this paper is organized as follows. Section 2 briefly introduces the statistical word alignment model. Section 3 describes parameter estimation method using the labeled data. Section 4 presents our semi-supervised boosting method. Section 5 reports the experimental results. Finally, we conclude in section 6.

2 Statistical Word Alignment Model

According to the IBM models (Brown et al., 1993), the statistical word alignment model can be generally represented as in equation (1).

$$\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{e})=\frac{\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{e})}{\sum_{\mathbf{a}'}\Pr(\mathbf{a}',\mathbf{f}\mid\mathbf{e})} \quad (1)$$

Where **e** and **f** represent the source sentence and the target sentence, respectively.

In this paper, we use a simplified IBM model 4 (Al-Onaizan et al., 1999), which is shown in equation (2). This simplified version does not take into account word classes as described in Brown et al. (1993).

$$\begin{aligned}\Pr(\mathbf{a},\mathbf{f}\mid\mathbf{e})=&\binom{m-\phi_0}{\phi_0}p_0^{m-2\phi_0}p_1^{\phi_0}\cdot\\&\prod_{i=1}^{l}n(\phi_i\mid e_i)\cdot\prod_{j=1}^{m}t(f_j\mid e_{a_j})\cdot\\&\Big(\prod_{j=1,a_j\neq 0}^{m}([j=h(a_j)]\cdot d_1(j-c_{\rho_{a_j}}))+\\&\prod_{j=1,a_j\neq 0}^{m}([j\neq h(a_j)]\cdot d_{>1}(j-p(j)))\Big)\end{aligned} \quad (2)$$

l, m are the lengths of the source sentence and the target sentence respectively.

j is the position index of the target word.

a_j is the position of the source word aligned to the j^{th} target word.

ϕ_i is the number of target words that e_i is aligned to.

p_0, p_1 are the fertility probabilities for e_0, and $p_0+p_1=1$.

$t(f_j\mid e_{a_j})$ is the word translation probability.

$n(\phi_i\mid e_i)$ is the fertility probability.

$d_1(j-c_{\rho_{a_j}})$ is the distortion probability for the head word of cept[1] *i*.

$d_{>1}(j-p(j))$ is the distortion probability for the non-head words of cept *i*.

$h(i)=\min_k\{k: i=a_k\}$ is the head of cept *i*.

$p(j)=\max_{k<j}\{k: a_j=a_k\}$.

ρ_i is the first word before e_i with non-zero fertility.

c_i is the center of cept *i*.

3 Parameter Estimation with Labeled Data

With the labeled data, instead of using EM algorithm, we directly estimate the three main parameters in model 4: translation probability, fertility probability, and distortion probability.

[1] A cept is defined as the set of target words connected to a source word (Brown et al., 1993).

3.1 Translation Probability

The translation probability is estimated from the labeled data as described in (3).

$$t(f_j \mid e_i) = \frac{count(e_i, f_j)}{\sum_{f'} count(e_i, f')} \quad (3)$$

Where $count(e_i, f_j)$ is the occurring frequency of e_i aligned to f_j in the labeled data.

3.2 Fertility Probability

The fertility probability $n(\phi_i \mid e_i)$ describes the distribution of the numbers of words that e_i is aligned to. It is estimated as described in (4).

$$n(\phi_i \mid e_i) = \frac{count(\phi_i, e_i)}{\sum_{\phi'} count(\phi', e_i)} \quad (4)$$

Where $count(\phi_i, e_i)$ describes the occurring frequency of word e_i aligned to ϕ_i target words in the labeled data.

p_0 and p_1 describe the fertility probabilities for e_0. And p_0 and p_1 sum to 1. We estimate p_0 directly from the labeled data, which is shown in (5).

$$p_0 = \frac{\#\,Aligned - \#\,Null}{\#\,Aligned} \quad (5)$$

Where $\#\,Aligned$ is the occurring frequency of the target words that have counterparts in the source language. $\#\,Null$ is the occurring frequency of the target words that have no counterparts in the source language.

3.3 Distortion Probability

There are two kinds of distortion probability in model 4: one for head words and the other for non-head words. Both of the distortion probabilities describe the distribution of relative positions Thus, if we let $\Delta j_1 = j - c_{\rho_i}$ and $\Delta j_{>1} = j - p(j)$, the distortion probabilities for head words and non-head words are estimated in (6) and (7) with the labeled data, respectively.

$$d_1(\Delta j_1) = \frac{\sum_{j, c_{\rho_i}} \delta(\Delta j_1, j - c_{\rho_i})}{\sum_{\Delta j_1'} \sum_{j', c_{\rho_{i'}}} \delta(\Delta j_1', j' - c_{\rho_{i'}})} \quad (6)$$

$$d_{>1}(\Delta j_{>1}) = \frac{\sum_{j, p(j)} \delta(\Delta j_{>1}, j - p(j))}{\sum_{\Delta j_{>1}'} \sum_{j', p(j')} \delta(\Delta j_{>1}', j' - p(j'))} \quad (7)$$

Where $\delta(x, y) = 1$ if $x = y$. Otherwise, $\delta(x, y) = 0$.

4 Boosting with Labeled Data and Unlabeled Data

In this section, we first propose a semi-supervised AdaBoost algorithm for word alignment, which uses both the labeled data and the unlabeled data. Based on the semi-supervised algorithm, we describe two boosting methods for word alignment. And then we develop a method to combine the results of the two boosting methods.

4.1 Semi-Supervised AdaBoost Algorithm for Word Alignment

Figure 1 shows the semi-supervised AdaBoost algorithm for word alignment by using labeled and unlabeled data. Compared with the supervised Adaboost algorithm, this semi-supervised AdaBoost algorithm mainly has five differences.

Word Alignment Model

The first is the word alignment model, which is taken as a learner in the boosting algorithm. The word alignment model is built using both the labeled data and the unlabeled data. With the labeled data, we train a supervised model by directly estimating the parameters in the IBM model as described in section 3. With the unlabeled data, we train an unsupervised model using the same EM algorithm in Brown et al. (1993). Then we build an interpolation model by linearly interpolating these two word alignment models, which is shown in (8). This interpolated model is used as the model M_l described in figure 1.

$$\begin{aligned} &\Pr(\mathbf{a}, \mathbf{f} \mid \mathbf{e}) \\ &= \lambda \cdot \Pr_S(\mathbf{a}, \mathbf{f} \mid \mathbf{e}) + (1 - \lambda) \cdot \Pr_U(\mathbf{a}, \mathbf{f} \mid \mathbf{e}) \end{aligned} \quad (8)$$

Where $\Pr_S(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$ and $\Pr_U(\mathbf{a}, \mathbf{f} \mid \mathbf{e})$ are the trained supervised model and unsupervised model, respectively. λ is an interpolation weight. We train the weight in equation (8) in the same way as described in Wu et al. (2005).

Pseudo Reference Set for Unlabeled Data

The second is the reference set for the unlabeled data. For the unlabeled data, we automatically build a *pseudo* reference set. In order to build a reliable pseudo reference set, we perform bi-directional word alignment on the training data using the interpolated model trained on the first round. Bi-directional word alignment includes alignment in two directions (source to

Input: A training set S_T including m bilingual sentence pairs;
The reference set R_T for the training data;
The reference sets R_L and R_U ($R_L, R_U \subseteq R_T$) for the labeled data S_L and the unlabeled data S_U respectively, where $S_T = S_U \cup S_L$ and $S_U \cap S_L = \text{NULL}$;
A loop count L.

(1) Initialize the weights:
$w_1(i) = 1/m, \quad i = 1,...,m$
(2) For $l = 1$ to L, execute steps (3) to (9).
(3) For each sentence pair i, normalize the weights on the training set:
$p_l(i) = w_l(i) / \sum_j w_l(j), \quad i = 1,...,m$
(4) Update the word alignment model M_l based on the weighted training data.
(5) Perform word alignment on the training set with the alignment model M_l:
$h_l = M_l(p_l)$
(6) Calculate the error of h_l with the reference set R_L: $\varepsilon_l = \sum_i p_l(i) \cdot \alpha(i)$
Where $\alpha(i)$ is calculated as in equation (9).
(7) If $\varepsilon_l > 1/2$, then let $L = l - 1$, and end the training process.
(8) Let $\beta_l = \varepsilon_l / (1 - \varepsilon_l)$.
(9) For all i, compute new weights:
$w_{l+1}(i) = w_l(i) \cdot (k + (n-k) \cdot \beta_l) / n$
where, n represents n alignment links in the i^{th} sentence pair. k represents the number of error links as compared with R_T.

Output: The final word alignment result for a source word e:

$$h_F(e) = \arg\max_f RS(e, f) = \arg\max_f \sum_{l=1}^{L} (\log \frac{1}{\beta_l}) \cdot WT_l(e, f) \cdot \delta(h_l(e), f)$$

Where $\delta(x, y) = 1$ if $x = y$. Otherwise, $\delta(x, y) = 0$. $WT_l(e, f)$ is the weight of the alignment link (e, f) produced by the model M_l, which is calculated as described in equation (10).

Figure 1. The Semi-Supervised Adaboost Algorithm for Word Alignment

target and target to source) as described in Och and Ney (2000). Thus, we get two sets of alignment results A_1 and A_2 on the unlabeled data. Based on these two sets, we use a modified "refined" method (Och and Ney, 2000) to construct a pseudo reference set R_U.

(1) The intersection $I = A_1 \cap A_2$ is added to the reference set R_U.
(2) We add $(e, f) \in A_1 \cup A_2$ to R_U if a) is satisfied or both b) and c) are satisfied.
   a) Neither e nor f has an alignment in R_U and $p(f|e)$ is greater than a threshold δ_1.
   $$p(f|e) = \frac{count(e, f)}{\sum_{f'} count(e, f')}$$
   Where $count(e, f)$ is the occurring frequency of the alignment link (e, f) in the bi-directional word alignment results.
   b) (e, f) has a horizontal or a vertical neighbor that is already in R_U.
   c) The set $R_U \cup (e, f)$ does not contain alignments with both horizontal and vertical neighbors.

Error of Word Aligner

The third is the calculation of the error of the individual word aligner on each round. For word alignment, a sentence pair is taken as a sample. Thus, we calculate the error rate of each sentence pair as described in (9), which is the same as described in Wu and Wang (2005).

$$\alpha(i) = 1 - \frac{2|S_W \cap S_R|}{|S_W| + |S_R|} \quad (9)$$

Where S_W represents the set of alignment links of a sentence pair i identified by the individual interpolated model on each round. S_R is the reference alignment set for the sentence pair.

With the error rate of each sentence pair, we calculate the error of the word aligner on each round. Although we build a pseudo reference set R_U for the unlabeled data, it contains alignment errors. Thus, the weighted sum of the error rates of sentence pairs in the labeled data instead of that in the entire training data is used as the error of the word aligner.

Weights Update for Sentence Pairs

The forth is the weight update for sentence pairs according to the error and the reference set. In a sentence pair, there are usually several word alignment links. Some are correct, and others may be incorrect. Thus, we update the weights according to the number of correct and incorrect alignment links as compared with the reference set, which is shown in step (9) in figure 1.

Weights for Word Alignment Links

The fifth is the weights used when we construct the final ensemble. Besides the weight $\log(1/\beta_l)$, which is the confidence measure of the l^{th} word aligner, we also use the weight $WT_l(e, f)$ to measure the confidence of each alignment link produced by the model M_l. The weight $WT_l(e, f)$ is calculated as shown in (10). Wu and Wang (2005) proved that adding this weight improved the word alignment results.

$$WT_l(e, f) = \frac{2 \times count(e, f)}{\sum_{f'} count(e, f') + \sum_{e'} count(e', f)} \quad (10)$$

Where $count(e, f)$ is the occurring frequency of the alignment link (e, f) in the word alignment results of the training data produced by the model M_l.

4.2 Method 1

This method only uses the labeled data as training data. According to the algorithm in figure 1, we obtain $S_T = S_L$ and $R_T = R_L$. Thus, we only change the distribution of the labeled data. However, we build an unsupervised model using the unlabeled data. On each round, we keep this unsupervised model unchanged, and we rebuild the supervised model by estimating the parameters as described in section 3 with the weighted training data. Then we interpolate the supervised model and the unsupervised model to obtain an interpolated model as described in section 4.1. The interpolated model is used as the alignment model M_l in figure 1. Thus, in this interpolated model, we use both the labeled and unlabeled data. On each round, we rebuild the interpolated model using the rebuilt supervised model and the unchanged unsupervised model. This interpolated model is used to align the training data.

According to the reference set of the labeled data, we calculate the error of the word aligner on each round. According to the error and the reference set, we update the weight of each sample in the labeled data.

4.3 Method 2

This method uses both the labeled data and the unlabeled data as training data. Thus, we set $S_T = S_L \cup S_U$ and $R_T = R_L \cup R_U$ as described in figure 1. With the labeled data, we build a supervised model, which is kept unchanged on each round.[2] With the weighted samples in the training data, we rebuild the unsupervised model with EM algorithm on each round. Based on these two models, we built an interpolated model as described in section 4.1. The interpolated model is used as the alignment model M_l in figure 1. On each round, we rebuild the interpolated model using the unchanged supervised model and the rebuilt unsupervised model. Then the interpolated model is used to align the training data.

Since the training data includes both labeled and unlabeled data, we need to build a pseudo reference set R_U for the unlabeled data using the method described in section 4.1. According to the reference set R_L of the labeled data, we calculate the error of the word aligner on each round. Then, according to the pseudo reference set R_U and the reference set R_L, we update the weight of each sentence pair in the unlabeled data and in the labeled data, respectively.

There are four main differences between Method 2 and Method 1.

(1) On each round, Method 2 changes the distribution of both the labeled data and the unlabeled data, while Method 1 only changes the distribution of the labeled data.
(2) Method 2 rebuilds the unsupervised model, while Method 1 rebuilds the supervised model.
(3) Method 2 uses the labeled data instead of the entire training data to estimate the error of the word aligner on each round.
(4) Method 2 uses an automatically built pseudo reference set to update the weights for the sentence pairs in the unlabeled data.

4.4 Combination

In the above two sections, we described two semi-supervised boosting methods for word alignment. Although we use interpolated models

[2] In fact, we can also rebuild the supervised model according to the weighted labeled data. In this case, as we know, the error of the supervised model increases. Thus, we keep the supervised model unchanged in this method.

for word alignment in both Method 1 and Method 2, the interpolated models are trained with different weighted data. Thus, they perform differently on word alignment. In order to further improve the word alignment results, we combine the results of the above two methods as described in (11).

$$h_{3,\mathrm{F}}(e) = \arg\max_{f}(\lambda_1 \cdot RS_1(e, f) + \lambda_2 \cdot RS_2(e, f)) \quad (11)$$

Where $h_{3,\mathrm{F}}(e)$ is the combined hypothesis for word alignment. $RS_1(e, f)$ and $RS_2(e, f)$ are the two ensemble results as shown in figure 1 for Method 1 and Method 2, respectively. λ_1 and λ_2 are the constant weights.

5 Experiments

In this paper, we take English to Chinese word alignment as a case study.

5.1 Data

We have two kinds of training data from general domain: Labeled Data (LD) and Unlabeled Data (UD). The Chinese sentences in the data are automatically segmented into words. The statistics for the data is shown in Table 1. The labeled data is manually word aligned, including 156,421 alignment links.

Data	# Sentence Pairs	# English Words	# Chinese Words
LD	31,069	255,504	302,470
UD	329,350	4,682,103	4,480,034

Table 1. Statistics for Training Data

We use 1,000 sentence pairs as testing set, which are not included in LD or UD. The testing set is also manually word aligned, including 8,634 alignment links in the testing set[3].

5.2 Evaluation Metrics

We use the same evaluation metrics as described in Wu et al. (2005), which is similar to those in (Och and Ney, 2000). The difference lies in that Wu et al. (2005) take all alignment links as sure links.

If we use S_{G} to represent the set of alignment links identified by the proposed method and S_{C} to denote the reference alignment set, the methods to calculate the precision, recall, f-measure, and alignment error rate (AER) are shown in equations (12), (13), (14), and (15). It can be seen that the higher the f-measure is, the lower the alignment error rate is.

$$precision = \frac{|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{G}}|} \quad (12)$$

$$recall = \frac{|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{C}}|} \quad (13)$$

$$fmeasure = \frac{2\times|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{G}}|+|S_{\mathrm{C}}|} \quad (14)$$

$$AER = 1 - \frac{2\times|S_{\mathrm{G}} \cap S_{\mathrm{C}}|}{|S_{\mathrm{G}}|+|S_{\mathrm{C}}|} = 1 - fmeasure \quad (15)$$

5.3 Experimental Results

With the data in section 5.1, we get the word alignment results shown in table 2. For all of the methods in this table, we perform bi-directional (source to target and target to source) word alignment, and obtain two alignment results on the testing set. Based on the two results, we get the "refined" combination as described in Och and Ney (2000). Thus, the results in table 2 are those of the "refined" combination. For EM training, we use the GIZA++ toolkit[4].

Results of Supervised Methods

Using the labeled data, we use two methods to estimate the parameters in IBM model 4: one is to use the EM algorithm, and the other is to estimate the parameters directly from the labeled data as described in section 3. In table 2, the method "Labeled+EM" estimates the parameters with the EM algorithm, which is an unsupervised method without boosting. And the method "Labeled+Direct" estimates the parameters directly from the labeled data, which is a supervised method without boosting. "Labeled+EM+Boost" and "Labeled+Direct+Boost" represent the two supervised boosting methods for the above two parameter estimation methods.

Our methods that directly estimate parameters in IBM model 4 are better than that using the EM algorithm. "Labeled+Direct" is better than "Labeled+EM", achieving a relative error rate reduction of 22.97%. And "Labeled+Direct+Boost" is better than "Labeled+EM+Boost", achieving a relative error rate reduction of 22.98%. In addition, the two boosting methods perform better than their corresponding methods without

[3] For a non one-to-one link, if *m* source words are aligned to *n* target words, we take it as one alignment link instead of *m***n* alignment links.

[4] It is located at http://www.fjoch.com/ GIZA++.html.

Method	Precision	Recall	F-Measure	AER
Labeled+EM	0.6588	0.5210	0.5819	0.4181
Labeled+Direct	0.7269	0.6609	0.6924	0.3076
Labeled+EM+Boost	0.7384	0.5651	0.6402	0.3598
Labeled+Direct+Boost	0.7771	0.6757	0.7229	0.2771
Unlabeled+EM	0.7485	0.6667	0.7052	0.2948
Unlabeled+EM+Boost	0.8056	0.7070	0.7531	0.2469
Interpolated	0.7555	0.7084	0.7312	0.2688
Method 1	0.7986	0.7197	0.7571	0.2429
Method 2	0.8060	0.7388	0.7709	0.2291
Combination	0.8175	0.7858	0.8013	0.1987

Table 2. Word Alignment Results

boosting. For example, "Labeled+Direct+Boost" achieves an error rate reduction of 9.92% as compared with "Labeled+Direct".

Results of Unsupervised Methods

With the unlabeled data, we use the EM algorithm to estimate the parameters in the model. The method "Unlabeled+EM" represents an unsupervised method without boosting. And the method "Unlabeled+EM+Boost" uses the same unsupervised Adaboost algorithm as described in Wu and Wang (2005).

The boosting method "Unlabeled+EM+Boost" achieves a relative error rate reduction of 16.25% as compared with "Unlabeled+EM". In addition, the unsupervised boosting method "Unlabeled+EM+Boost" performs better than the supervised boosting method "Labeled+Direct+Boost", achieving an error rate reduction of 10.90%. This is because the size of labeled data is too small to subject to data sparseness problem.

Results of Semi-Supervised Methods

By using both the labeled and the unlabeled data, we interpolate the models trained by "Labeled+Direct" and "Unlabeled+EM" to get an interpolated model. Here, we use "interpolated" to represent it. "Method 1" and "Method 2" represent the semi-supervised boosting methods described in section 4.2 and section 4.3, respectively. "Combination" denotes the method described in section 4.4, which combines "Method 1" and "Method 2". Both of the weights λ_1 and λ_2 in equation (11) are set to 0.5.

"Interpolated" performs better than the methods using only labeled data or unlabeled data. It achieves relative error rate reductions of 12.61% and 8.82% as compared with "Labeled+Direct" and "Unlabeled+EM", respectively.

Using an interpolation model, the two semi-supervised boosting methods "Method 1" and "Method 2" outperform the supervised boosting method "Labeled+Direct+Boost", achieving a relative error rate reduction of 12.34% and 17.32% respectively. In addition, the two semi-supervised boosting methods perform better than the unsupervised boosting method "Unlabeled+EM+Boost". "Method 1" performs slightly better than "Unlabeled+EM+Boost". This is because we only change the distribution of the labeled data in "Method 1". "Method 2" achieves an error rate reduction of 7.77% as compared with "Unlabeled+EM+Boost". This is because we use the interpolated model in our semi-supervised boosting method, while "Unlabeled+EM+Boost" only uses the unsupervised model.

Moreover, the combination of the two semi-supervised boosting methods further improves the results, achieving relative error rate reductions of 18.20% and 13.27% as compared with "Method 1" and "Method 2", respectively. It also outperforms both the supervised boosting method "Labeled+Direct+Boost" and the unsupervised boosting method "Unlabeled+EM+Boost", achieving relative error rate reductions of 28.29% and 19.52% respectively.

Summary of the Results

From the above result, it can be seen that all boosting methods perform better than their corresponding methods without boosting. The semi-supervised boosting methods outperform the supervised boosting method and the unsupervised boosting method.

6 Conclusion and Future Work

This paper proposed a semi-supervised boosting algorithm to improve statistical word alignment with limited labeled data and large amounts of unlabeled data. In this algorithm, we built an interpolated model by using both the labeled data

and the unlabeled data. This interpolated model was employed as a learner in the algorithm. Then, we automatically built a pseudo reference for the unlabeled data, and calculated the error rate of each word aligner with the labeled data. Based on this algorithm, we investigated two methods for word alignment. In addition, we developed a method to combine the results of the above two semi-supervised boosting methods.

Experimental results indicate that our semi-supervised boosting method outperforms the unsupervised boosting method as described in Wu and Wang (2005), achieving a relative error rate reduction of 19.52%. And it also outperforms the supervised boosting method that only uses the labeled data, achieving a relative error rate reduction of 28.29%. Experimental results also show that all boosting methods outperform their corresponding methods without boosting.

In the future, we will evaluate our method with an available standard testing set. And we will also evaluate the word alignment results in a machine translation system, to examine whether lower word alignment error rate will result in higher translation accuracy.

References

Yaser Al-Onaizan, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky. 1999. Statistical Machine Translation Final Report. *Johns Hopkins University Workshop.*

Sugato Basu, Mikhail Bilenko, and Raymond J. Mooney. 2004. Probabilistic Framework for Semi-Supervised Clustering. In *Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004)*, pages 59-68.

Avrim Blum and Tom Mitchell. 1998. Combing Labeled and Unlabeled Data with Co-training. In *Proc. of the 11th Conference on Computational Learning Theory (COLT-1998)*, pages1-10.

Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19(2): 263-311.

Colin Cherry and Dekang Lin. 2003. A Probability Model to Improve Word Alignment. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-2003)*, pages 88-95.

Michael Collins and Yoram Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proc. of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-1999)*, pages 100-110.

Thomas G. Dietterich. 2000. Ensemble Methods in Machine Learning. In *Proc. of the First International Workshop on Multiple Classifier Systems (MCS-2000)*, pages 1-15.

Yoav Freund and Robert E. Schapire. 1996. Experiments with a New Boosting Algorithm. In *Proc. of the 13th International Conference on Machine Learning (ICML-1996)*, pages 148-156.

Franz Josef Och and Hermann Ney. 2000. Improved Statistical Alignment Models. In *Proc. of the 38th Annual Meeting of the Association for Computational Linguistics (ACL-2000)*, pages 440-447.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19-51.

Thanh Phong Pham, Hwee Tou Ng, and Wee Sun Lee 2005. Word Sense Disambiguation with Semi-Supervised Learning. In *Proc. of the 20th National Conference on Artificial Intelligence (AAAI 2005)*, pages 1093-1098.

Anoop Sarkar. 2001. Applying Co-Training Methods to Statistical Parsing. In *Proc. of the 2nd Meeting of the North American Association for Computational Linguistics(NAACL-2001)*, pages 175-182.

Joachims Thorsten. 1999. Transductive Inference for Text Classification Using Support Vector Machines. In *Proc. of the 16th International Conference on Machine Learning (ICML-1999)*, pages 200-209.

Dekai Wu. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics*, 23(3): 377-403.

Hua Wu and Haifeng Wang. 2005. Boosting Statistical Word Alignment. In *Proc. of the 10th Machine Translation Summit*, pages 313-320.

Hua Wu, Haifeng Wang, and Zhanyi Liu. 2005. Alignment Model Adaptation for Domain-Specific Word Alignment. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 467-474.

David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-1995)*, pages 189-196.

Hao Zhang and Daniel Gildea. 2005. Stochastic Lexicalized Inversion Transduction Grammar for Alignment. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*, pages 475-482.