Investigating Problems of Semi-Supervised Learning for Word Sense Disambiguation

Anh-Cuong Le, Akira Shimazu, and Le-Minh Nguyen

School of Information Science
Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa, 923-1292, JAPAN

Abstract. Word Sense Disambiguation (WSD) is the problem of determining the right sense of a polysemous word in a given context. In this paper, we will investigate the use of unlabeled data for WSD within the framework of semi supervised learning, in which the original labeled dataset is iteratively extended by exploiting unlabeled data. This paper addresses two problems occurring in this approach: determining a subset of new labeled data at each extension and generating the final classifier. By giving solutions for these problems, we generate some variants of bootstrapping algorithms and apply to word sense disambiguation. The experiments were done on the datasets of four words: *interest*, *line*, *hard*, and *serve*; and on English lexical sample of Senseval-3.

1 Introduction

Word sense disambiguation involves the association of a given word in a text or discourse with a particular sense among numerous potential senses of that word. For this task, many supervised machine learning algorithms have been used for the WSD task, including Naïve Bayes, decision trees, an exemplar-based, support vector machines, maximum entropy, etc (see, e.g., [3]). However, supervised methods require large labeled data for training, which are expensive to obtain. Therefore, many researchers have recently concentrated their efforts on how to use unlabeled data to boost the performance of supervised learning for WSD, such as in [5, 11, 10, 7]. The process of using both labeled and unlabeled data for training is called *semi-supervised learning* or *bootstrapping*.

In this paper, we focus on an approach that iteratively enlarges labeled data with new labeled examples which are obtained from unlabeled data. A common method for the extension of labeled data is to use the classifier trained on the current labeled dataset to detect labels for unlabeled examples. Among those new labeled examples, some highly accurate ones are selected and added to the current labeled dataset. This process is iteratively repeated until there is no unlabeled example left, or until the number of iterations reaches a pre-defined number. Two well-known methods of this approach are self-training [9] and co-training [1]. A general algorithm of this approach is sketched in Fig. 1 (it can be considered as the general self-training algorithm). We address two problems occurring in this algorithm as follows.

Input:
L (labeled data); U (unlabeled data); h (a supervised learning algorithm)
$k = 0$; K is the maximum number of iteration
Output:
L_{out} is the final set of labeled data
or $h(L_{out})$ is the final classifier
Algorithm:
Repeat
1. $k \leftarrow k + 1$
2. use h and L to train a classifier $h(L)$
3. use $h(L)$ to label U, and obtain a labeled set U_L
4. get $L' \subset U_L$ consisting of high confident examples
5. $L \leftarrow L \cup L'$; $U \leftarrow U \setminus L'$

Until $U = \emptyset$ or $k > K$
6. $L_{out} \leftarrow L$

Fig. 1. A General Bootstrapping Algorithm

P_1: The first problem is how to determine a subset of new labeled examples at each extension of labeled data. To obtain "good" new labeled examples, we consider two criteria: the correctness of labeling and the number of new labeled examples. It is clear that adding a large number of misclassified examples into the labeled dataset will probably result in decreasing the accuracy of the classifier, and increasing confidence [1] of labeling may receive a small number of new labeled examples. Suppose that we have a new example which is assigned a label with a detection probability, previous studies normally use a threshold for this probability to decide whether a new labeled example will be selected or not, such as in [9, 1]. However, choosing a high threshold will create difficulty in extending labeled data, and does not always result in correct classification. On the contrary, choosing a lower threshold may result in more misclassified examples. In this paper, we will increase confidence of new labeled examples by using one more supervised classifier for the detection process.

P_2: The second problem is that of how to generate the final classifier when the process of extending labeled data is completed. According to the framework in Fig. 1, this process will be stopped when the number of iterations reaches a pre-specified value, or until the unlabeled dataset becomes empty. Normally, the classifier, which is built on the labeled data obtained at the last iteration, is chosen as the final classifier. Some studies use a development dataset to find the most appropriate value for the number of iterations, such as in [7, 5]. Others used an upper bound of error rate of training data as the condition for stopping this process, such as in [2]. However, the last classifier may be built based on new training data with some misclassified examples (related to problem P_1), so some advantages and some disadvantages are concurrently brought to the last classifier in comparison with the initial supervised classifier (i.e. the classifier

[1] The term "increasing confidence" of a classifier means using any method, by that we hope to increase the accuracy of label detection of this classifier

which trained on the original labeled dataset). In our knowledge, this observation, which has not been observed in previous studies yet, may lead us to a solution of combining the advantages of both the initial and the last classifiers under classifier combination strategies.

The solutions for these two problems consequently generate variants of bootstrapping algorithms. They were evaluated through experiments on the four words *interest*, *line*, *hard*, *server*; and on English lexical sample of Senseval-3. Therefore, next section presents proposed solutions and new bootstrapping algorithms. Section 3 presents experiments with data preparation, results and discussion. Finally, conclusions are presented in section 4.

2 Problems and Solutions

2.1 Extending labeled data

This section presents problem P_1. An usual approach to this task is using a supervised learning algorithm to train a classifier based on the labeled dataset, and then using this classifier to detect labels for examples in a subset U' of the current unlabeled dataset U. Suppose h is the supervised classifier and $l(h, e)$ is the label of example e detected by h with probability $P(h, e)$. If $P(h, e)$ is greater than a threshold α, then example e with new label $l(h, e)$ will be considered to be added to L. As mentioned previously, using a classifier with threshold α for determining new labeled examples may cause a tradeoff problem between the extensibility and the accuracy of label detection. Furthermore, increasing α does not always result in an increase of accuracy in new labeled examples. To tackle this problem, we propose a solution that uses one more classifier, h' [2], as follows.

Suppose that at each extension, the maximum number of new labeled examples which are added to the current labeled dataset is N. We firstly use h and h' to detect labels for examples in U', and then select the examples e such that $P(h, e) \geq \alpha$. Among such new labeled examples, the examples which have agreements of labelling of h and h' are selected. If there exist more than N such examples, we will prefer the examples with high confidence of detection of h. In addition, if there are not enough agreements of labelling between h and h', we use h to select more new labeled examples, and also prefer the examples e with high $P(h, e)$. By this solution we can increase the confidence of new labeled examples, and also maintain the capability of extending labeled data.

Concerning this task, we also build a procedure of retaining class distribution. It is necessary because if a classifier is built based on the training data with a bias in some classes (i.e. some classes dominate others), then this bias will be increased at each extension of the labeled data. This problem is also considered in previous studies such as [1, 8]. For a set of labeled examples, we divide it into the subsets such that the examples in each subset have the same label. We

[2] we assume that the two classifiers, h and h', are unbiased and the errors made by them would not fully overlap

call them class-based subsets. So that, building a procedure of retaining class distribution for a set of new labeled examples means to resize its class-based subsets to keep class distribution of the original labeled dataset.

2.2 Generating final classifier

There is a fact that when extending labeled data, the feature space will be extended concurrently with adding some examples which are misclabeled. Therefore, the quality of labelling of the last classifier (trained on the last labeled dataset) will depend on each particular test example. In comparison with the initial supervised classifier, the final classifier may be better in detecting some test examples if these examples contain many new features covering by the new labeled examples. In the contrary case, if test examples can be well labeled by the initial supervised classifier, it is not necessary and risk to use the last classifier to label these examples. A natural way to utilize advantages of both these classifiers is to combine them when making decision of labelling. It then becomes a classifier combination problem. Based on OWA combination rules, which were also used for WSD, as presented in [4], we found that the *median* rule and *max* rule are intuitively applicable for this objective. These combination rules are recalled as follows.

For each example e, suppose $P_i(h_1, e),\ i = 1, \ldots, m$ is the probability distribution on class set $\{c_1, \ldots, c_m\}$ of using classifier h_1 to label e. A similar definition is applied to $P_i(h_2, e)$, for classifier h_2 and $i = 1, \ldots, m$.

Combining h_1 and h_2 using median and max rules, we obtain new probability distributions as follows:

$$P_i^{median}(h_1, h_2, e) = (P_i(h_1, e) + P_i(h_2, e))/2,\ i = 1, \ldots, m$$

$$P_i^{max}(h_1, h_2, e) = max\{P_i(h_1, e),\ P_i(h_2, e)\},\ i = 1, \ldots, m$$

Then the class c_k will be assigned to example e when using median rule (or max rule) iff:

$$P_k^{median}(h_1, h_2, e) \geq P_i^{median}(h_1, h_2, e), \forall i = 1, \ldots, m$$

$$P_k^{max}(h_1, h_2, e) \geq P_i^{max}(h_1, h_2, e), \forall i = 1, \ldots, m$$

2.3 A new bootstrapping algorithm

By the solutions as mentioned above, we generate a new bootstrapping algorithm as shown in Fig. 2. In this algorithm, $Resize(L_1, N)$ is the procedure for retaining class distribution, which returns new sizes for all class-based subsets of L_1 and satisfy that sum of all new sizes is less than or equal N; L is the labeled data; U is the unlabeled data; A and A' are two supervised learning algorithms, in which A is the primary one; α is a threshold; K is the maximum number of

iteration; N is the maximum number of added labeled examples at each iteration; $C = \{c_1, \ldots, c_m\}$ is the set of classes; suppose S is a set of labeled examples, define $C_i(S) = \{e | e \in S,\ l(e) = c_i\}$, where $l(e)$ is the label of example e, and $i = 1, \ldots, m$.

Input: L, U, A, A', α, K, N.
Output: classifier H
Algorithm:
initial: $k \leftarrow 0$; $L_0 \leftarrow L$; create a pool $U' \in U$ (randomly selected)
Repeat
1. $k \leftarrow k + 1$; $L_1 \leftarrow \emptyset$; $L_2 \leftarrow \emptyset$
2. training A and A' on L and generate classifiers h and h', respectively
3. *for each* example $e \in U'$ *do*
   if $P(h, e) \geq \alpha$ *then* add e with new label $l(h, e)$ to L_1
4. *Resize*(L_1, N)
   obtaining n_i which is the new size of subset $C_i(L_1)$, for $i = 1, \ldots, m$
5. *for each* class $c_i \in C$ *do*
   5.1 sort examples of $C_i(L_1)$ in descending order, following the function $f(e)$:
   if $l(h, e) = l(h', e)$ *then* $f(e) \leftarrow 1 + P(h, e)$ *else* $f(e) \leftarrow P(h, e)$
   5.2. add first n_i examples of $C_i(L_1)$ to L_2
6. $L \leftarrow L \cup L_2$; rebuild U' from U and $L_1 \setminus L_2$
Until $k > K$ or $|L_2| = 0$
7. Let $A(L_0)$ and $A(L)$ be classifiers obtained by train A on L_0 and L, respectively then return H is the combination between $A(L_0)$ and $A(L)$

Fig. 2. A New Bootstrapping Algorithm

In fact, the new algorithm is an extension of the general self-training by providing solutions for problems P_1 and P_2. For experiment, we consider four variants of the bootstrapping algorithm as follows: A_1 is the general self-training algorithm; A_2 is the self-training algorithm with the solution for problem P_2 (we separate this by A_{2a} and A_{2b} respect to median and max rules for the combination step); A_3 is the self-training algorithm with the solution for problem P_1; A_4 is the self-training algorithm with the solutions for problem P_1 and P_2 (i.e. the new algorithm; we separate this by A_{4a} and A_{4b} respect to median and max rules for the combination step).

3 Experiment

3.1 Data

The first experiment was carried out on the datasets of the four words, namely *interest*, *line*, *serve*, and *hard*, which were obtained from Pedersen's homepage[3]. All examples in these datasets were tagged with the right senses. The sizes of

[3] http://www.d.umn.edu/~tpederse/data.html

these data are 2369, 4143, 4378, and 4342 for *interest*, *line*, *serve*, and *hard*, respectively. These datasets are large enough for dividing into labeled and unlabeled data sets. Furthermore, because we knew the tagged senses of examples in unlabeled dataset, we could evaluate the correctness of the new labeled examples (for problem P_1). We randomly extract 100 examples for labeled data, 300 examples for test data, and the remaining examples are treated as unlabeled examples.

The second test was carried out on English lexical sample from Senseval-3 data[4]. Unlabeled data in this experiment was collected from the British National Corpus (BNC) with about 3000 examples for each ambiguous word. Note that because the English lexical sample was also retrieved from BNC, so for a fair test, we removed all contexts from unlabeled dataset which also appear in the training or test datasets.

3.2 Feature selection

Two of the most important kinds of information for determining the senses of an ambiguous word are the topic of the context and the relational information representing the structural relations between the target word and the surrounding words in a local context. A bag of unordered words in the context can represent the topic of the context, while collocation can represent grammatical information as well as the order relations between the target word and neighboring words. We also use more information about local context represented by words assigned with their positions, and their part-of-speech assigned with positions. These kinds of features are investigated in many WSD studies such as [6, 5]. Particulary, all features used in our experiment fall in the kinds: a set of content words that include nouns, verbs, and adjectives, in a context window size 50; a set of collocations of words (we selected a set of collocations as presented in [3]); a set of words assigned with their positions in a window size 5; and a set of part-of-speech tags of these word also assigned with their positions, also in window size 5.

3.3 Parameters and Results

For the new algorithm, we chose naive Bayes (NB) as the primary supervised learning algorithm and support vector machines (SVM) as the second algorithm. Because SVM is a discriminative learning algorithm while NB is a generative one, so this difference will make SVM as an independent and confident classifier to verify the correctness of NB classifier's detection.

For the remaining parameters, we fix the maximum size of unlabeled dataset used at each iteration $|U'| = 800$ and the maximum size of new labeled examples which are added to the labeled dataset $N = 300$. The number of iteration K runs from 1 to 15 when testing on datasets of the four words, and then the best was used for testing on Senseval-3.

[4] http://www.senseval.org/senseval3

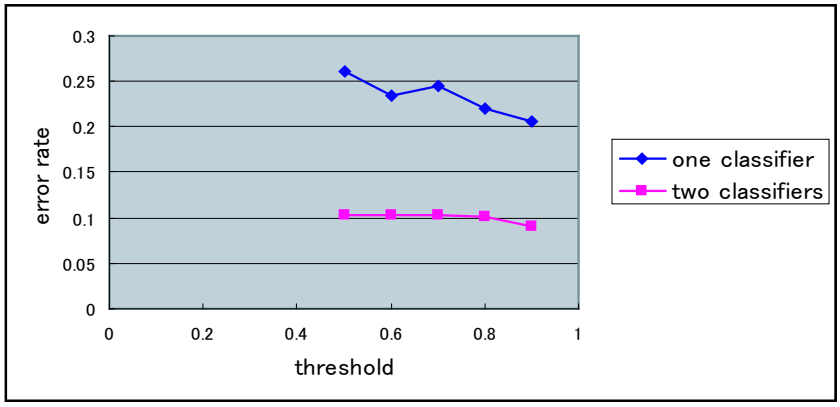

Fig. 3. Test problem P_1 using one classifier and two classifiers

Table 1. Test on Senseval-3.

Algorithms		Average accuracy(%)
NB		70.13
SVM		69.45
A1 (self-training)		70.6
A2	median rule (A_{2a})	71.80
(self-training with P_2 solution)	max rule (A_{2b})	71.76
A3 (self-training with P_1 solution)		71.03
A4	median rule (A_{4a})	72.07
(self-training with P_1 and P_2 solutions)	max rule (A_{4b})	71.97

In the first experiment, we investigated the problem P_1 through various values of α, and the effectiveness of using one classifier (NB) and two classifiers (NB and SVM). The threshold α received values in $\{0.5, 0.6, 0.7, 0.8, 0.9\}$. The result in Fig. 3 shows that using one more classifier much decrease error rate (from about 25% to 10%). In addition, this result also suggests the choice of $\alpha = 0.6$, which may ensure both the correctness and the extendibility of labeled data.

Table 1 shows a test on English lexical sample of Senseval-3 [5] where we ran the bootstrapping algorithms 10 times and got the average. The algorithm A_3 is better than the algorithm A_1 (71.03% in comparison with 70.6%) shows the effectiveness of problem P_1. In addition, it should be noted that the algorithm with solution for problem P_2 still improves much the accuracy of self-training in the case not using solution for problem P_1 (the algorithm A_2). It can be explained that: permitting a high error rate of new labeled examples will expand more feature space which results in recognizing more new examples, and the proposed combination strategies keep its advantage and decrease its disadvantage.

[5] We used the recall measure of fine-grained scoring for the evaluation.

In summary, combining the solutions for both problems P_1 and P_2 gives the best result, and it increases about 1.9% of accuracy in comparison with supervised learning using NB.

4 Conclusion

In this paper, we have showed two problems of semi-supervised learning, particularly for self-training algorithm. They include the problem of extending labeled data and that of generating the final classifier. These problems have been investigated in WSD problem, and corresponding solutions were given. For the first problem, we used one more classifier to decrease error rate of new labeled examples. And for the second problem, we used two strategies of classifier combination including median and max rules to utilize both advantages of the last classifier (built based on the extended labeled data) and the initial supervised classifier. With those solutions, a new bootstrapping algorithm with several variants were generated. The experiments show that the proposed solutions are effective for improving semi-supervised learning. In addition, it also showed that unlabeled data significantly improves supervised learning in word sense disambiguation.

Acknowledgement

This research is partly conducted as a program for the "Fostering Talent in Emergent Research Fields" in Special Coordination Funds for Promoting Science and Technology by the Japanese Ministry of Education, Culture, Sports, Science and Technology.

References

1. Blum A., and Mitchell T., 1998. Combining labeled and unlabeled data with co-training. In *Proc. COLT*, pages 92–100.
2. Goldman S. and Zhou Y., 2000. Enhancing supervised learning with unlabeled data. In *Proc. ICML*, pages 327–334, 2000.
3. Lee Y. K., and Ng H. T., 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *Proc. EMNLP*, pages 41–48.
4. Le C. A., Huynh V-N, Dam H-C, Shimazu A., 2005. Combining Classifiers Based on OWA Operators with an Application to Word Sense Disambiguation. In *Proc. RSFDGrC*, Vol 1, pages 512–521.
5. Mihalcea R., 2004. Co-training and Self-training for Word Sense Disambiguation. In *Proc. CoNLL*, pages 33–40.
6. Ng H. T. and Lee H. B., 1996. Integrating multiple knowledge sources to Disambiguate Word Sense: An exemplar-based approach. In *Proc. ACL*, pages 40–47.
7. Pham T. P., Ng H. T., and Lee W. S., 2005. Word Sense Disambiguation with Semi-Supervised Learning. In *Proc. AAAI*, pages 1093–1098.
8. Pierce D., and Cardie C., 2001. Limitations of co-training for natural language learning from large datasets. In *Proc. EMNLP*, pages 1–9.

9. Yarowsky D., 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. ACL*, pages 189–196.
10. Yu N. Z., Hong J. D., and Lim T. C., 2005. Word Sense Disambiguation Using Label Propagation Based Semi-supervised Learning Method. In *Proc. ACL*, pages. 395–402.
11. SU W., CARPUAT M., and WU D., 2004. Semi-Supervised Training of a Kernel PCA-Based Model for Word Sense Disambiguation. In *Proc. COLING*, pages 1298–1304.