

Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning

Zheng-Yu Niu, Dong-Hong Ji

Institute for Infocomm Research

21 Heng Mui Keng Terrace

119613 Singapore

{zniu, dhji}@i2r.a-star.edu.sg

Chew-Lim Tan

Department of Computer Science

National University of Singapore

3 Science Drive 2

117543 Singapore

tancl@comp.nus.edu.sg

Abstract

Shortage of manually sense-tagged data is an obstacle to supervised word sense disambiguation (WSD) methods. In this paper we investigate a label propagation based semi-supervised learning algorithm for WSD, which combines unlabeled data with labeled data in learning process by representing labeled and unlabeled examples as vertices in a weighted graph and iteratively propagating the label information from any vertex to nearby vertices until this process converges. This label propagation process realizes a global consistency assumption: similar examples should have similar labels. Our experimental results on benchmark corpora indicate that it consistently outperforms SVM when only very few labeled examples are available, and its performance is also better than monolingual bootstrapping, and comparable to bilingual bootstrapping.

1 Introduction

In this paper, we address the problem of word sense disambiguation (WSD), which is to assign an appropriate sense to an occurrence of a word in a given context. Many methods have been proposed to deal with this problem (Ide and Véronis, 1998), including supervised learning algorithms (Leacock et al., 1998; Towel and Voorheest, 1998), semi-supervised learning algorithms (Dagan and Itai, 1994; Yarowsky, 1995), and unsupervised learning algorithms (Pedersen and Bruce, 1997; Schütze, 1998).

Supervised sense disambiguation has been very successful, but it requires a lot of manually sense-tagged data. Fully unsupervised methods do not need the definition of senses and manually sense-tagged data, but the sense clustering result can not be directly used in many NLP tasks since there is no sense label for each instance in clusters. Considering both the availability of a large amount of unlabelled data and direct usage of word senses, semi-supervised learning has received great attention recently.

Semi-supervised methods for WSD are characterized in terms of exploiting unlabeled data in learning process. Bootstrapping is a commonly used semi-supervised learning algorithm for WSD. In each iteration step of bootstrapping, unlabeled examples are classified using a model learned from labeled data. In other words, it is based on a local consistency assumption: examples close to labeled examples within same class will have same labels, which is also the assumption underlying many supervised learning algorithms, such as kNN.

Recently a promising family of semi-supervised learning algorithms are introduced, which can effectively combine unlabeled data with labeled data in learning process by exploiting manifold structure (cluster structure) in data (Belkin and Niyogi, 2002; Blum and Chawla, 2001; Blum et al., 2004; Joachims, 2003; Szummer and Jaakkola, 2001; Zhou et al., 2003; Zhu and Ghahramani, 2002; Zhu et al., 2003). Such methods perform classification based on a global consistency assumption: similar examples should have similar labels. In other words, the labels of unlabeled examples are determined by considering not only the similarity between labeled and unlabeled examples, but also the similarity between unlabeled examples.

Here we investigate a label propagation based semi-supervised learning algorithm (LP algorithm) (Zhu and Ghahramani, 2002) for WSD, which works by representing labeled and unlabeled examples as vertices in a connected graph, then propagating the label information from any vertex to nearby vertices through weighted edges, finally inferring the labels of unlabeled examples after the propagation process converges.

Compared with bootstrapping, LP algorithm is based on a global consistency assumption. Intuitively, if there is at least one labeled example in each cluster that consists of similar examples, then the unlabeled examples will have the same labels as the labeled examples on the same cluster by propagating the label information of any example to nearby examples according to their proximity.

This paper is organized as follows. Firstly we will formulate WSD problem in the context of semi-supervised learning in section 2. Then in section 3 we will describe the LP

algorithm and discuss the difference between a supervised learning algorithm (SVM), bootstrapping algorithm and LP algorithm in detail. Section 4 will give out the experimental results of LP algorithm on widely used benchmark corpora. Section 5 will be devoted to a brief review of related work on semi-supervised learning for WSD. Finally we will conclude our work and suggest possible improvement in section 6.

2 Problem Setup

Let $X = \{x_i\}_{i=1}^n$, a set of contexts of occurrences of an ambiguous word w , where x_i represents the context of the i -th occurrence, and n is the total number of this word's occurrences. Let $S = \{s_j\}_{j=1}^c$, which denotes the sense tag set of w . The first l examples $x_g (1 \leq g \leq l)$ are labeled as $y_g (y_g \in S)$ and the remaining $u (l + u = n)$ examples $x_h (l + 1 \leq h \leq n)$ are unlabeled. The goal is to predict the sense of w in context x_h by the use of label information from x_g and the similarity among examples in X .

The manifold structure in dataset X can be represented as a connected graph, where each vertex corresponds to an example, and the edge between any two examples x_i and x_j is weighted so that the closer the vertices in some distance measure, the larger the weight associated with this edge. The weights are defined as follows: $W_{ij} = \exp(-\frac{d_{ij}^2}{\sigma^2})$ if $i \neq j$ and $W_{ii} = 0 (1 \leq i, j \leq n)$, where d_{ij} is the distance (ex. Euclidean distance) between x_i and x_j , and σ is used to control the weight W_{ij} .

3 Semi-Supervised Learning Method

3.1 Label Propagation Algorithm

In label propagation algorithm (Zhu and Ghahramani, 2002), the label information of any vertex in the graph is propagated to nearby vertices through weighted edges until a global stable stage is achieved. Larger edge weights allow labels to travel through easier. Thus the closer the examples, more likely they have similar labels (the global consistency assumption).

In label propagation process, the label distribution of initial labeled data is clamped in each iteration to replenish the label sources from these labeled data. Thus the labeled data act like sources to push out labels through unlabeled data. With this push from labeled examples, the class boundaries will be pushed through edges with large weights and settle in gaps along edges with low weights. If the data structure fits the classification goal, then LP algorithm can use these unlabeled data to help learning classification plane.

Let $Y^0 \in N^{n \times c}$ represent initial soft labels attached to each vertex, where $Y_{ij}^0 = 1$ if y_i is s_j and 0 otherwise. Let Y_L^0 be the top l rows of Y^0 and Y_U^0 be the remaining u rows. Y_L^0 is consistent with the labeling in labeled data, and the initialization of Y_U^0 can be arbitrary.

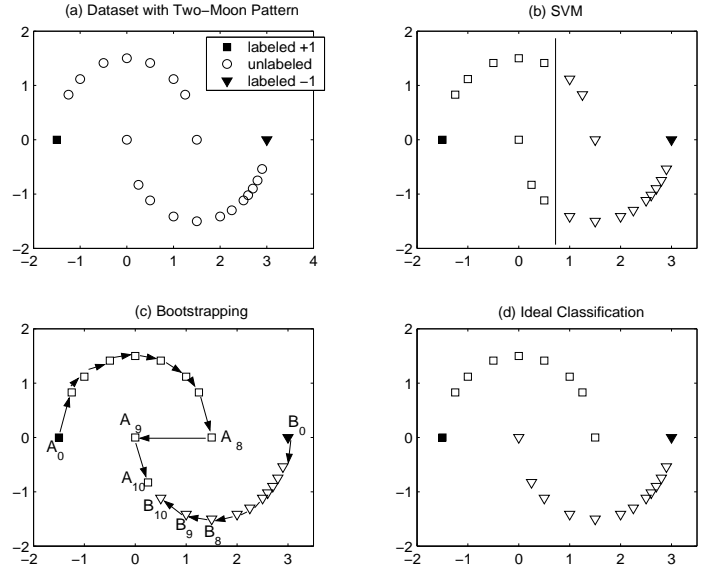


Figure 1: Classification result on two-moon pattern dataset. (a) two-moon pattern dataset with two labeled points, (b) classification result by SVM, (c) labeling procedure of bootstrapping algorithm, (d) ideal classification.

Optimally we expect that the value of W_{ij} across different classes is as small as possible and the value of W_{ij} within same class is as large as possible. This will make label propagation to stay within same class. In this paper, we set σ as the average distance between labeled examples from different classes.

Define $n \times n$ probability transition matrix $T_{ij} = P(j \rightarrow i) = \frac{W_{ij}}{\sum_{k=1}^n W_{kj}}$, where T_{ij} is the probability to jump from example x_j to example x_i .

Compute the row-normalized matrix \bar{T} by $\bar{T}_{ij} = T_{ij} / \sum_{k=1}^n T_{ik}$. This normalization is to maintain the class probability interpretation of Y .

Then LP algorithm is defined as follows:

1. Initially set $t=0$, where t is the iteration index;
2. Propagate the label by $Y^{t+1} = \bar{T}Y^t$;
3. Clamp the labeled data by replacing the top l row of Y^{t+1} with Y_L^0 . Repeat from step 2 until Y^t converges;
4. Assign $x_h (l + 1 \leq h \leq n)$ with a label $y_h = \text{argmax}_j Y_{hj}$.

This algorithm has been shown to converge to a unique solution, which is $\hat{Y}_U = \lim_{t \rightarrow \infty} Y_U^t = (I - \bar{T}_{uu})^{-1} \bar{T}_{ul} Y_L^0$ (Zhu and Ghahramani, 2002). We can see that this solution can be obtained without iteration and the initialization of Y_U^0 is not important, since Y_U^0 does not affect the estimation of \hat{Y}_U . I is $u \times u$ identity matrix. \bar{T}_{uu} and \bar{T}_{ul} are acquired by splitting matrix \bar{T} after the l -th row and the l -th column into 4 sub-matrices.

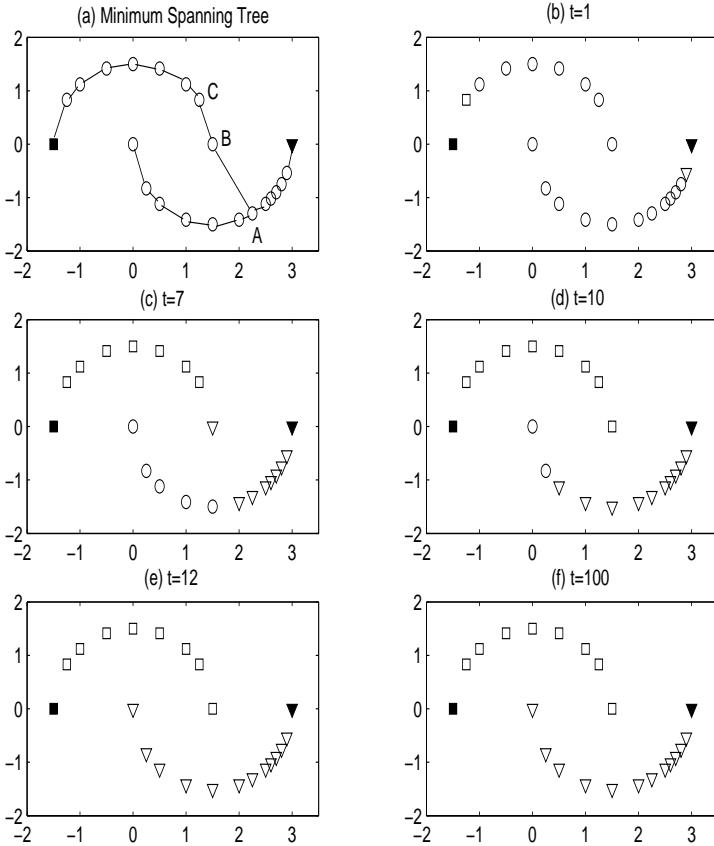


Figure 2: Classification result of LP on two-moon pattern dataset. (a) minimum spanning tree of this dataset. The convergence process of LP algorithm with t varying from 1 to 100 is shown from (b) to (f).

3.2 Comparison between SVM, Bootstrapping and LP

For WSD, SVM is one of the state of the art supervised learning algorithms (Mihalcea et al., 2004), while bootstrapping is one of the state of the art semi-supervised learning algorithms (Li and Li, 2004; Yarowsky, 1995). For comparing LP with SVM and bootstrapping, let us consider a dataset with two-moon pattern shown in Figure 1(a). The upper moon consists of 9 points, while the lower moon consists of 13 points. There is only one labeled point in each moon, and the remaining 20 points are unlabeled. The distance measure is Euclidian distance. We can see that points in one moon should be more similar to each other than the points across the moons.

Figure 1(b) shows the classification result of SVM. The vertical line denotes the classification hyperplane, which has the maximum separating margin with respect to the labeled points in two classes. We can find that SVM does not work well when labeled data can not reveal the structure (the moon pattern) in each class. The reason is that the classification hyperplane was learned only from labeled data. In other words, the coherent structure (the two-moon pattern) in unlabeled data was not explored when inferring class boundary.

Let us consider the bootstrapping algorithm. Figure 1(c)

shows the bootstrapping procedure using kNN ($k=1$) as base classifier with user-specified parameter $b = 1$ (the number of added examples from unlabeled data into classified data for each class in each iteration). Each arrow in Figure 1(c) represents one iteration for each class. After eight iterations of bootstrapping, $A_1 \sim A_8$ were tagged as $+1$, and $B_1 \sim B_8$ were tagged as -1 , while $A_9 \sim A_{10}$ and $B_9 \sim B_{10}$ were still untagged. Then at the ninth iteration, A_9 was tagged as $+1$ since the label of A_9 was determined only by tagged points in kNN model: A_9 is closer to any point in $\{A_0 \sim A_8\}$ than to any point in $\{B_0 \sim B_8\}$, regardless of the intrinsic structure in data: $A_9 \sim A_{10}$ and $B_9 \sim B_{10}$ are closer to points in lower moon than to points in the upper moon. In other words, bootstrapping method uses the unlabeled data under a local consistency based strategy. This is the reason that two points A_9 and A_{10} are misclassified (shown in Figure 1(c)).

The class distribution is usually fixed during the iteration procedure to avoid degenerate solutions, which is controlled by b . But the class distribution can not be reliably estimated when only very few labeled examples are available. Another parameter α (the threshold of classification confidence) can be used for avoiding misclassification. For example, only unlabeled points with Euclidian distance from labeled points less than the value of radius of the moon should be added into classified data. But this value of α is usually difficult to acquire. If classification confidence is set too large (very low value of α), many points will be left in remaining unlabeled data after bootstrapping. It will limit the improvement of resulting classifier’s performance.

From above analysis we find that (1) Both SVM and bootstrapping are based on a local consistency assumption. (2) The two parameters b and α in bootstrapping are not easy to specify in practice.

For simplification of illustration, we ran LP algorithm on a connected graph-minimum spanning tree generated for this dataset, shown in Figure 2(a). A, B, C represent three points, and edge $A - B$ connects the two moons. Figure 2(b)- 2(f) shows the convergence process of LP algorithm with t increasing from 1 to 100.

When $t = 1$, the label information of labeled data was pushed to only nearby points. After seven iteration steps ($t = 7$), point B in upper moon was misclassified as -1 since it firstly received label information from point A through the edge connecting two moons. But after another three iteration steps ($t=10$), this misclassified point was re-tagged as $+1$. The reason is that with the push of label information from nearby points, the value of $Y_{B,+1}$ was higher than $Y_{B,-1}$. In other words, the weight of edge $B - C$ is larger than that of edge $B - A$, which makes the $+1$ label of point C to travel to point B easier. Finally, when $t \geq 12$ the LP algorithm converged to a fixed point, which achieved the ideal classification result.

4 Experiments and Results

4.1 Experiment Design

For empirical comparison with SVM and bootstrapping, we evaluated LP algorithm on widely used benchmark corpora - “interest”, “line”¹ and the data in English lexical sample task of Senseval-3 (including all 57 English words)².

We used three types of features to capture contextual information: part-of-speech of neighboring words with position information, unordered single words in topical context, and local collocations (the same as the feature set used in (Lee and Ng, 2002) except that we did not use syntactic relations). For SVM, we did not perform feature selection since feature selection deteriorated its performance (Lee and Ng, 2002). For LP algorithm, we employed a simple feature selection method on the three datasets: All the features with occurrence frequency less than 3 times in dataset were removed when converting contexts into feature vectors.

In this paper, we investigated two distance measures: cosine similarity and Jensen-Shannon (JS) divergence (Lin, 1991). Cosine similarity measures the angle between two feature vectors, while JS divergence measures the distance between two probability distributions if each feature vector is considered as probability distribution over features.

Let $JS(p, q)$ represent JS divergence between probability distribution $p(f)$ and $q(f)$ (f is a random variable), which is defined as

$$JS(p, q) = \pi_p D_{KL}(p||\bar{p}) + \pi_q D_{KL}(q||\bar{p}), \quad (1)$$

$$D_{KL}(p||\bar{p}) = \sum_f p \log \frac{p}{\bar{p}}, \quad (2)$$

$$D_{KL}(q||\bar{p}) = \sum_f q \log \frac{q}{\bar{p}}, \quad (3)$$

$$\bar{p} = \pi_p p + \pi_q q, \quad (4)$$

where π_p and π_q are prior probabilities, and $\pi_p, \pi_q > 0, \pi_p + \pi_q = 1$.

4.2 Experiment 1: LP vs. SVM

In this experiment, we evaluated LP algorithm and SVM algorithm³ on the data of English lexical sample task in Senseval-3. This dataset consists of fully labeled training set and test set. We use l examples from training set as labeled data, and the remaining training examples and all the test examples as unlabeled data. For each labeled set size l , we performed 20 trials. In each trial, we randomly sampled l labeled examples for each word from training set. If any sense was absent from the sampled labeled set, we redid

Table 1: Average accuracies over 20 trials and paired t-test result of SVM and LP algorithm on Senseval-3 corpus with percentage of training set increasing from 1% to 100%.

Percentage	SVM	LP_{cosine}	LP_{JS}
1%	25.7%	30.1%	30.8%
10%	54.5%	55.9%	56.5%
25%	63.8%	63.7%	64.9%
50%	68.4%	67.3%	68.6%
75%	70.5%	68.9%	70.3%
100%	71.6%	70.0%	71.8%

Percentage	SVM vs. LP_{cosine}		SVM vs. LP_{JS}	
	p-value	Sign.	p-value	Sign.
1%	1.54e-006	≪	1.86e-007	≪
10%	4.00e-007	≪	3.18e-010	≪
25%	5.92e-001	~	3.96e-007	≪
50%	3.25e-007	≫	1.39e-001	~
75%	1.32e-012	≫	4.71e-002	>
100%	-	-	-	-

the sampling. We conducted experiments with different values of l , including $1\% \times N_{w,train}$, $10\% \times N_{w,train}$, $25\% \times N_{w,train}$, $50\% \times N_{w,train}$, $75\% \times N_{w,train}$, $100\% \times N_{w,train}$ ($N_{w,train}$ is the number of examples in training set of word w). SVM and LP were evaluated using accuracy on test set of Senseval-3.

We conducted paired t-test on the accuracy figures for each value of l . Paired t-test was not run when percentage= 100%, since there was only one paired accuracy figures. Paired t-test is usually used to estimate the difference in means between normal populations based on a set of random paired observations. $\{\ll, \gg\}$, $\{<, >\}$, and \sim correspond to p-value ≤ 0.01 , $(0.01, 0.05]$, and > 0.05 respectively. \ll (or \gg) means that the performance of LP is significantly better (or significantly worse) than SVM. $<$ (or $>$) means that the performance of LP is better (or worse) than SVM. \sim means that the performance of LP is almost as same as SVM.

Table 1 reports the average accuracies and paired t-test result of SVM and LP with different sizes of labeled data.

From Table 1, we see that with small labeled dataset (percentage of labeled data $\leq 10\%$), LP algorithm performs significantly better than SVM. When the percentage of labeled data increases from 25% to 100%, the performance of LP_{JS} algorithm is still comparable to SVM.

4.3 Experiment 2: LP vs. Bootstrapping

In (Li and Li, 2004), they used “interest” and “line” corpora as test data. For the word “interest”, they used its four major senses. For comparison with their results, we took reduced “interest” corpus (constructed by retaining four major senses) and complete “line” corpus as evaluation data. In (Li and Li, 2004), c is the number of senses of ambiguous word, and b ($b = 15$) is the number of examples added into classified data for each class in each iteration of bootstrapping. $c \times b$ can be considered as the size of initial labeled data in their

¹ Available at <http://www.d.umn.edu/~tperdse/data.html>

² Available at <http://www.senseval.org/senseval3>

³ A tool, SVM^{light} , is used to perform classification, which can be downloaded at <http://svmlight.joachims.org/>. The kernel function we used is linear. Other parameters in SVM are set as default.

Table 2: Accuracies from (Li and Li, 2004) and average accuracies of LP with $c \times b$ labeled examples on “interest” and “line” corpora. Major is a baseline method in which they always choose the most frequent sense. MB-D denotes monolingual bootstrapping with decision list as base classifier, MB-B represents monolingual bootstrapping with ensemble of Naive Bayes as base classifier, and BB is bilingual bootstrapping with ensemble of Naive Bayes as base classifier.

Ambiguous words	Accuracies from (Li and Li, 2004)			
	Major	MB-D	MB-B	BB
interest	54.6%	54.7%	69.3%	75.5%
line	53.5%	55.6%	54.1%	62.7%

Ambiguous words	Our results		
	#labeled examples	LP_{cosine}	LP_{JS}
interest	$4 \times 15 = 60$	80.2%	79.8%
line	$6 \times 15 = 90$	60.3%	59.4%

bootstrapping algorithm. We ran LP algorithm with 20 trials on reduced “interest” corpus and complete “line” corpus. In each trial, we randomly sampled b labeled examples for each sense of “interest” or “line” as labeled data. The rest served as both unlabeled data and test data. Table 2 summarizes the average accuracies of LP algorithm on the two corpora.

Table 2 also lists the accuracies of monolingual bootstrapping algorithm (MB), bilingual bootstrapping algorithm (BB) on “interest” and “line” corpora. We can find that LP algorithm performs better than MB-D and MB-B on both “interest” and “line” corpora.

BB extends MB by constructing classifiers in two languages, classifying unclassified data in two languages and exchanging information regarding classified data between the two languages. From Table 2, we see that the performance of LP is still comparable to BB although LP does not refer to a second language monolingual corpus.

4.4 An Example: Word “use”

For investigating the reason for LP to outperform SVM and bootstrapping, we used the data of word “use” in Senseval-3 as an example (totally 26 examples in training set and 14 examples in test set). For visualization, we conducted unsupervised nonlinear dimensionality reduction⁴ on these 40 feature vectors with 210 dimensions. Figure 3 (a) shows the dimensionality reduced vectors in two-dimensional space. We randomly sampled only one labeled example for each sense of word “use” as labeled data. The remaining data in training set and test set served as unlabeled data for bootstrapping and LP. All of these three algorithms are evaluated using accuracy on test set.

⁴We use a tool, *Isomap*, to perform nonlinear dimensionality reduction by computing two-dimensional, 39-nearest-neighbor-preserving embedding of 210-dimensional input. This tool is available at <http://isomap.stanford.edu/>.

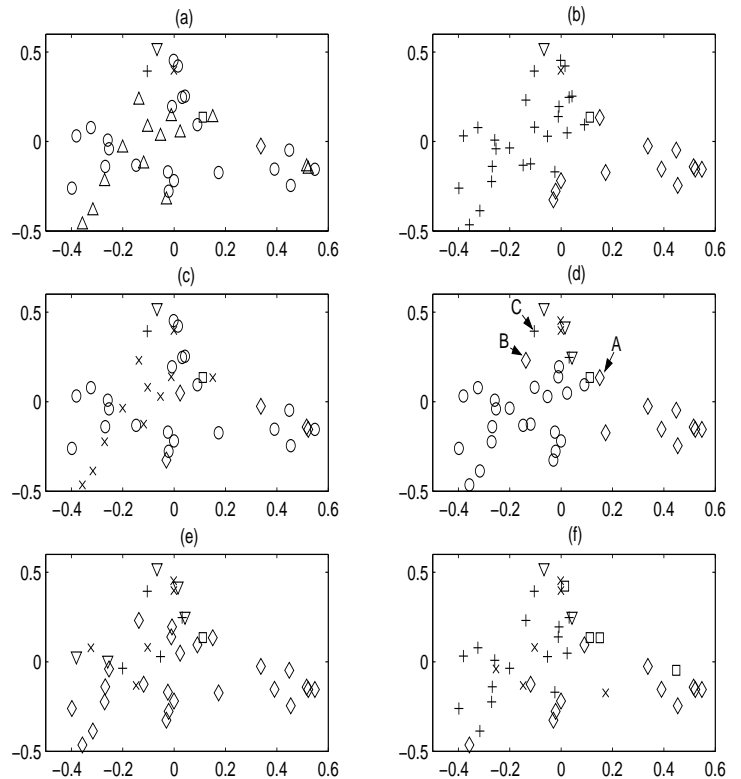


Figure 3: Comparison of sense disambiguation results between SVM, bootstrapping and LP on word “use”. (a) dataset with only one labeled point for each sense of word “use” before sense disambiguation (\circ and \triangle denote the unlabeled examples in training set and test set respectively, and other five symbols ($+$, \times , \square , \diamond , and ∇) represent the labeled examples with respective sense tag sampled from training set.). (b) ground-truth result, (c) classification result on test set by SVM (accuracy = $\frac{3}{14} = 21.4\%$), (d) classified data after bootstrapping, (e) classification result on both training set and test set by 1NN (accuracy = $\frac{6}{14} = 42.9\%$), (f) classification result on both training set and test set by LP (accuracy = $\frac{10}{14} = 71.4\%$).

From Figure 3 (c) we can see that SVM misclassified many examples from class $+$ into class \times , regardless of the intrinsic structure in data.

For comparison, we conducted an additional experiment of bootstrapping with kNN ($k=1$) as base classifier. The parameter b is set as 1. The other parameter α is set as the minimum distance between labeled examples with different sense tags. Then only unlabeled examples with the distance from nearest labeled example less than α will be added into classified data in each iteration. Firstly we ran bootstrapping with 1NN as base classifier on this dataset to augment initial labeled data. The resulting classified data is shown in Figure 3 (d). Then a 1NN model was learned on this classified data and we used this model to perform classification on the remaining unlabeled data. Figure 3 (e) reports the final classification result

by this 1NN model. We can see that bootstrapping does not perform well since it is susceptible to small noise in dataset. For example, in Figure 3 (d), the unlabeled example B^5 happened to be closest to labeled example A , then 1NN tagged example B with label \diamond . But the correct label of example B should be $+$ as shown in Figure 3 (b). This error caused misclassification of other unlabeled examples that should have label $+$.

In LP algorithm, the label information of example C can travel to B through unlabeled data. Then example A will compete with C and other unlabeled examples around B when determining the label of B . In other words, the label of unlabeled examples are determined not only by nearby labeled examples, but also by nearby unlabeled examples. Using this classification strategy achieves better performance than the local consistency based strategy adopted by SVM and bootstrapping.

4.5 Experiment 3: LP_{\cosine} vs. LP_{JS}

Table 3 summarizes the performance comparison between LP_{\cosine} and LP_{JS} on three datasets. We can see that on Senseval-3 corpus, LP_{JS} performs significantly better than LP_{\cosine} , but their performance is almost comparable on “interest” and “line” corpora. This observation motivates us to automatically select a distance measure which will boost the performance of LP algorithm on a given dataset.

Cross-validation on labeled data is a widely accepted model selection method for supervised learning. But for the setting of semi-supervised learning ($l \ll u$), the labeled data is usually not enough for reliable model selection. In (Szummer and Jaakkola, 2001; Zhu and Ghahramani, 2002; Zhu et al., 2003), they suggested a label entropy criterion $H(Y_U)$ for model selection, where Y is the label matrix learned by their semi-supervised algorithms. The intuition behind their method is that good parameters should result in confident labeling. Entropy on matrix W ($H(W)$) is a commonly used measure for unsupervised feature selection (Dash and Liu, 2000). Another possible criterion is to measure the entropy of $c \times c$ inter-class distance matrix D calculated on labeled data, denoted as $H(D)$. $D_{i,j}$ represents the average distance between the i -th class and the j -th class. In this paper, we investigated these three criteria $H(D)$, $H(W)$ and $H(Y_U)$ for model selection. The distance measure can be automatically selected by minimizing the average value of function $H(D)$, $H(W)$ or $H(Y_U)$ over 20 trials.

Let Q be the $M \times N$ matrix. Function $H(Q)$ can measure the entropy of matrix Q , which is defined as (Dash and Liu,

⁵In the two-dimensional space, example B is not the closest example to A . The reason is that: (1) A is not close to most of nearby examples around B , and B is also not close to most of nearby examples around A ; (2) we use $Isomap$ to maximally preserve the neighborhood information between any example and all other examples, which caused the loss of neighborhood information between a few examples for obtaining a globally optimal solution.

Table 3: Performance comparison between LP_{\cosine} and LP_{JS} and the results of three model selection criteria are reported in following two tables. $<$ (or $>$) means that the average value of function $H(Q_{\cosine})$ is lower (or higher) than $H(Q_{JS})$, and it will result in selecting cosine (or JS) as distance measure. Q_{\cosine} represents the matrix calculated using cosine similarity. Q_{JS} represents the matrix calculated using JS divergence. \surd and \times denote correct and wrong prediction results respectively, while \circ means that any prediction is acceptable.

Data	LP_{\cosine} vs. LP_{JS}	
	p-value	Significance
Senseval-3 (1%)	4.21e-004	\ll
Senseval-3 (10%)	4.02e-005	\ll
Senseval-3 (25%)	5.93e-010	\ll
Senseval-3 (50%)	8.87e-010	\ll
Senseval-3 (75%)	1.98e-012	\ll
Senseval-3 (100%)	-	-
interest	3.25e-002	$>$
line	8.09e-002	\sim

Data	$H(D)$	$H(W)$	$H(Y_U)$
	cos. vs. JS	cos. vs. JS	cos. vs. JS
Senseval-3 (1%)	$> (\surd)$	$> (\surd)$	$< (\times)$
Senseval-3 (10%)	$< (\times)$	$> (\surd)$	$< (\times)$
Senseval-3 (25%)	$< (\times)$	$> (\surd)$	$< (\times)$
Senseval-3 (50%)	$> (\surd)$	$> (\surd)$	$> (\surd)$
Senseval-3 (75%)	$> (\surd)$	$> (\surd)$	$> (\surd)$
Senseval-3 (100%)	$< (\circ)$	$> (\circ)$	$< (\circ)$
interest	$< (\surd)$	$> (\times)$	$< (\surd)$
line	$> (\circ)$	$> (\circ)$	$> (\circ)$

2000):

$$S_{i,j} = \exp(-\alpha * Q_{i,j}), \quad (5)$$

$$H(Q) = - \sum_{i=1}^M \sum_{j=1}^N (S_{i,j} \log S_{i,j} + (1 - S_{i,j}) \log (1 - S_{i,j})), \quad (6)$$

where α is positive constant. The possible value of α is $-\frac{\ln 0.5}{\bar{I}}$, where $\bar{I} = \frac{1}{MN} \sum_{i,j} Q_{i,j}$. S is introduced for normalization of matrix Q . For Senseval-3 data, we calculated an overall average score of $H(Q)$ by $\sum_w \frac{N_{w,test}}{\sum_w N_{w,test}} H(Q_w)$. $N_{w,test}$ is the number of examples in test set of word w . $H(D)$, $H(W)$ and $H(Y_U)$ can be obtained by replacing Q with D , W and Y_U respectively.

Table 3 reports the automatic prediction results of these three criteria.

From Table 3, we can find that using $H(W)$ can consistently select the optimal distance measure when the performance gap between LP_{\cosine} and LP_{JS} is very large (denoted by \ll or \gg). But $H(D)$ and $H(Y_U)$ fail to find the optimal distance measure when only very few labeled examples are available (percentage of labeled data $\leq 10\%$).

$H(W)$ measures the separability of matrix W . Higher value of $H(W)$ means that the distance measure decreases the separability of examples in full dataset. Then the boundary among manifolds is obscured, which makes it difficult for

LP algorithm to locate this boundary. Thus higher value of $H(W)$ results in worse performance of LP algorithm.

When labeled dataset is small, the distances between classes can not be reliably estimated, which results in unreliable indication of the separability of examples in full dataset. This is the reason that $H(D)$ performs poorly on Senseval-3 corpus when the percentage of labeled data is less than 25%.

For $H(Y_U)$, small labeled dataset can not reveal the structure in data, which may bias the estimation of Y_U . Then the confidence of labeling $H(Y_U)$ can not properly indicate the performance of LP. This may interpret the poor performance of $H(Y_U)$ on Senseval-3 when percentage $\leq 10\%$.

5 Related Work

In fact, semi-supervised learning approaches for WSD roughly fall into three categories according to what is used for supervision in learning process.

(1) Using external resources, e.g., thesaurus or lexicons, to disambiguate word senses or automatically generate sense-tagged corpus:

In (Yarowsky, 1992), they automatically generated non-perfect sense-tagged corpus by tagging each target word with the semantic categories in Roget's thesaurus.

McCarthy et al. (2004) proposed a method to find predominant senses of ambiguous words by the use of automatically acquired thesaurus and WordNet.

In (Seo et al., 2004) they used an unsupervised statistical model to determine preferred sense among WordNet relatives of an ambiguous word in a given context of an occurrence of the ambiguous word.

(2) Exploiting the differences between mapping of words to senses in different languages by the use of bilingual corpora (e.g. parallel corpora or untagged monolingual corpora in two languages):

Brown et al. (1991) proposed a flip-flop algorithm to automatically align parallel corpora, which resulted in sense disambiguation components.

In (Dagan and Itai, 1994), their algorithm disambiguated senses of an ambiguous word by translating the senses of the ambiguous word into difference words in target language and selecting the preferred word (or sense) using a statistical model and lexical constraint.

Diab and Resnik (2002) used WordNet to disambiguate a group of similar translations in target language that correspond to the same word in source language, then projected the sense tag between the two languages to generate parallel sense-tagged corpora.

In (Ng et al., 2003), they exploited manually word-aligned parallel corpora to generate sense-tagged training data for WSD.

Li and Li (2004) proposed a bilingual bootstrapping algorithm, which benefited from the asymmetric many-to-many

sense mapping relationship between words in two languages.

(3) Automatically augmenting sense-tagged data to overcome the bottleneck of acquisition of manually sense-tagged data:

In (Hearst, 1991), bootstrapping was applied to augment manually sense-tagged data by iteratively training a base classifier on classified data, using the resulting classifier to classify unlabeled data, and adding the most confidently tagged examples into classified data.

Yarowsky (1995) improved this bootstrapping method in two aspects: (a) they used automatically generated seed examples as initial labeled data; (b) they exploited a redundant view (one sense per discourse property) to improve the performance of bootstrapping.

There are other efforts for improving base classifier in bootstrapping: Park et al. (2000) used committee learning algorithm as base classifier, while in (Mihalcea, 2004), majority voting was employed as classification method.

Some of the algorithms mentioned above may cross different categories. For example, the work in (Diab and Resnik, 2002) also falls into the first category since WordNet was used for sense disambiguation, and the algorithm presented in (Li and Li, 2004) is an extension of monolingual bootstrapping technique.

In contrast with the approaches in the first and second category, our corpus based method does not need external resources, including WordNet, bilingual lexicon, aligned parallel corpora, and machine translation tool. Our LP based WSD method belongs to the third category, which benefits from initial labeled dataset and raw unannotated monolingual corpora that can be cheaply acquired. In contrast with the popular bootstrapping technique, LP algorithm exhibits a theoretical advantage, that it performs classification based on a global consistency assumption.

6 Conclusions and Future Work

In this paper we have investigated a label propagation based semi-supervised learning algorithm for word sense disambiguation, which is based on a global consistency assumption: similar examples should have similar labels. In other words, the labels of unlabeled examples are determined not only by nearby labeled examples, but also by nearby unlabeled examples. Our analysis and experimental results demonstrate the potential of this manifold learning based algorithm. It achieves better performance than supervised learning method (SVM) when only very few labeled examples are available, and its performance on monolingual corpora is also better than monolingual bootstrapping and comparable to bilingual bootstrapping. Finally we suggest an entropy based method to automatically identify a distance measure that can boost the performance of LP algorithm on a given dataset.

LP algorithm has intimate connections with random walk (Suzummer and Jaakkola, 2001), spectral clustering (Shi and Malik, 2000) and graph mincut (Blum and Chawla, 2001; Blum et al., 2004; Joachims, 2003). In the future we may extend the evaluation of LP algorithm and related algorithms using more datasets for WSD.

There are some possible improvements on this work. Currently, we employ a very simple feature selection method. In the future, we would like to investigate the entropy based criterion $H(W)$ or other unsupervised model selection method (Lange et al., 2002) for parameter learning, including feature selection and weight matrix learning.

References

- Belkin, M., & Niyogi, P.. 2002. Using Manifold Structure for Partially Labeled Classification. *Advances in Neural Information Processing Systems 15*.
- Blum, A., & Chawla, S.. 2001. Learning from Labeled and Unlabeled Data Using Graph Mincuts. *Proceedings of the 18th International Conference on Machine Learning*.
- Blum, A., Lafferty, J., Rwebangira, R., & Reddy, R.. 2004. Semi-Supervised Learning Using Randomized Mincuts. *Proceedings of the 21st International Conference on Machine Learning*.
- Brown P., Stephen, D.P., Vincent, D.P., & Robert, Mercer.. 1991. Word Sense Disambiguation Using Statistical Methods. *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*.
- Dagan, I. & Itai A.. 1994. Word Sense Disambiguation Using A Second Language Monolingual Corpus. *Computational Linguistics*, Vol. 20(4), pp. 563-596.
- Dash, M., & Liu, H.. 2000. Feature Selection for Clustering. *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*(pp. 110–121).
- Diab, M., & Resnik, P.. 2002. An Unsupervised Method for Word Sense Tagging Using Parallel Corpora. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*(pp. 255–262).
- Hearst, M.. 1991. Noun Homograph Disambiguation using Local Context in Large Text Corpora. *Proceedings of the 7th Annual Conference of the UW Centre for the New OED and Text Research: Using Corpora*, 24:1, 1–41.
- Ide, N. & Véronis, J.. 1998. Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24:1, 1–41.
- Joachims, T.. 2003. Transductive Learning via Spectral Graph Partitioning. *Proceedings of the 20th International Conference on Machine Learning*.
- Lange, T., Braun, M., Roth, V., & Buhmann, J. M. 2002. Stability-Based Model Selection. *Advances in Neural Information Processing Systems 15*.
- Leacock, C., Miller, G.A. & Chodorow, M.. 1998. Using Corpus Statistics and WordNet Relations for Sense Identification. *Computational Linguistics*, 24:1, 147–165.
- Lee, Y.K. & Ng, H.T.. 2002. An Empirical Evaluation of Knowledge Sources and Learning Algorithms for Word Sense Disambiguation. *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, (pp. 41-48).
- Li, H. & Li, C.. 2004. Word Translation Disambiguation Using Bilingual Bootstrapping. *Computational Linguistics*, 30(1), 1-22.
- Lin, J. 1991. Divergence Measures Based on the Shannon Entropy. *IEEE Transactions on Information Theory*, 37:1, 145–150.
- McCarthy, D., Koeling, R., Weeds, J., & Carroll, J.. 2004. Finding Predominant Word Senses in Untagged Text. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*.
- Mihalcea R.. 2004. Co-training and Self-training for Word Sense Disambiguation. *Proceedings of the Conference on Natural Language Learning*.
- Mihalcea R., Chklovski, T., & Kilgariff, A.. 2004. The Senseval-3 English Lexical Sample Task. *Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*.
- Ng, H.T., Wang, B., & Chan, Y.S.. 2003. Exploiting Parallel Texts for Word Sense Disambiguation: An Empirical Study. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pp. 455-462.
- Park, S.B., Zhang, B.T., & Kim, Y.T.. 2000. Word Sense Disambiguation by Learning from Unlabeled Data. *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*.
- Pedersen, T., & Bruce, R.. 1997. Distinguishing Word Senses in Untagged Text. *Proceedings of the 2nd Conference on Empirical Methods in Natural Language Processing*, pp. 197–207.
- Schütze, H.. 1998. Automatic Word Sense Discrimination. *Computational Linguistics*, 24:1, 97–123.
- Seeger, M.. 2001. Learning with Labeled and Unlabeled Data. *Technical Report*, University of Edinburgh.
- Seo, H.C., Chung, H.J., Rim, H.C., Myaeng, S.H., & Kim, S.H.. 2004. Unsupervised Word Sense Disambiguation Using WordNet Relatives. *Computer, Speech and Language*, 18:3, 253–273.
- Shi, J., & Malik, J.. 2000. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 888-905.
- Suzummer, M., & Jaakkola, T.. 2001. Partially Labeled Classification with Markov Random Walks. *Advances in Neural Information Processing Systems 14*.
- Towel, G. & Voorheest, E.M.. 1998. Disambiguating Highly Ambiguous Words. *Computational Linguistics*, 24:1, 125–145.
- Yarowsky, D.. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189-196.
- Yarowsky, D.. 1992. Word Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. *Proceedings of the 14th International Conference on Computational Linguistics*, pp. 454-460.
- Zhou D., Bousquet, O., Lal, T.N., Weston, J., & Schölkopf, B.. 2003. Learning with Local and Global Consistency. *Advances in Neural Information Processing Systems 16*, pp. 321-328.
- Zhu, X. & Ghahramani, Z.. 2002. Learning from Labeled and Unlabeled Data with Label Propagation. *CMU CALD tech report CMU-CALD-02-107*.
- Zhu, X., Ghahramani, Z., & Lafferty, J.. 2003. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. *Proceedings of the 20th International Conference on Machine Learning*.