IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 12, Issue 4 (Jul. - Aug. 2013), PP 63-68
www.iosrjournals.org

Review: Semi-Supervised Learning Methods for Word Sense Disambiguation

Ms. Ankita Sati
(M.Tech Student, Banasthali Vidyapith, Banasthali, Jaipur, India)

Abstract : *Word sense disambiguation (WSD) is an open problem of natural language processing, which governs the process of identifying the appropriate sense of a word in a sentence, when the word has multiple meanings. Many approaches have been proposed to solve the problem, of which supervised learning approaches are the most successful. However supervised machine learning are limited by the difficulties in defining sense tags, acquiring labeled data for training and cannot utilize raw un-annotated data. Semi-supervised learning has recently become active research area because it require small amount of labeled training data and sometimes able to improve performance using unlabeled data. In this paper, we discuss the methods of semi-supervised learning and their performance.*
Keywords: *-word sense disambiguation, supervised learning, semi-supervised learning.*

Submitted date 19 June 2013 **Accepted Date: 24 June 2013**

I. Introduction

In a language, a word can have many different meanings, or senses. For example, *bank* in English can either mean a financial institution, or a sloping raised land. The task of word sense disambiguation (WSD) is to assign the correct sense to such ambiguous words based on the surrounding context. Word sense disambiguation (WSD) is a central question in the computational linguistics community. It is fundamental to natural language understanding and once it is improved, many other language processing tasks will benefit from it, such as information retrieval (IR) and machine translation (MT) (Ide and Veronis, 1998). Many methods have been proposed to deal with this problem, including supervised learning algorithms (Leacock et al., 1998), semi-supervised learning algorithms (Yarowsky, 1995), and unsupervised learning algorithms (Schutze, 1998).

Among these methods supervised sense disambiguation has been very successful, but it requires a lot of manually sense tagged data (labeled data) and cannot utilize raw unannotated data (unlabeled data) that can be cheaply acquired. Mostly, to achieve good performance, the amount of training data required by supervised learning is quite large. This is undesirable as hand-labeled training data is expensive and only available for a small set of words. Fully unsupervised methods do not need the definition of senses and manually sense-tagged data, but their sense clustering results cannot be directly used in many NLP tasks since there is no sense tag for each instance in clusters. Considering both the availability of a large amount of unlabeled data and direct use of word semi-supervised learning has recently become an active research area. It requires only a small amount of labeled training data and is sometimes able to improve performance using unlabeled data.

II. Semi–Supervised Learning

Semi-supervised learning is a class of machine learning techniques that use both labeled and unlabeled data for training - typically a small amount of labeled data with a large amount of unlabeled data. The name "semi-supervised learning" comes from the fact that the data used is between supervised learning (with completely labeled training data) and unsupervised learning (without any labeled training data). Many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce considerable improvement in learning accuracy. The acquisition of labeled data for a learning problem often requires a skilled human agent and the process is expensive and time consuming whereas acquisition of unlabeled data is relatively inexpensive and less time consuming.

As in the supervised learning framework, we are given a set of l independently identically distributed examples $x_1, \ldots\ldots, x_l \in X$ with corresponding labels $y_1, \ldots, y_l \in Y$. Additionally; we are given u unlabeled examples $x_{l+1}, \ldots, x_{l+u} \in X$. Semi-supervised learning attempts to make use of this combined information to surpass the classification performance that could be obtained either by discarding the unlabeled data and doing supervised learning or by discarding the labels and doing unsupervised learning.

Semi-supervised learning promises higher accuracies with less annotating effort. It is therefore of great theoretic and practical interest.

III. Semi–Supervised Learning Techniques

The semi-supervised or minimally supervised methods are gaining popularity because of their ability to get by with only a small amount of annotated reference data while often outperforming totally unsupervised methods on large data sets.

Semi-supervised methods for WSD are characterized in terms of exploiting unlabeled data in learning procedure with the requirement of predefined sense inventory for target words. They roughly fall into three categories according to what is used for supervision in learning process:

Using external resources, e.g., thesaurus or lexicons, to disambiguate word senses or automatically generate sense-tagged corpus, (Lesk, 1986; Lin, 1997; McCarthy et al., 2004; Yarowsky, 1992)

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; Dagan and Itai, 1994; Diab and Resnik, 2002; Li and Li, 2004; Ng et al., 2003)

Bootstrapping sense tagged seed examples to overcome the bottleneck of acquisition of large sense-tagged data (Hearst, 1991; Karov and Edelman, 1998; Mihalcea, 2004; Park et al., 2000; Yarowsky, 1995).

Bootstrapping algorithm is commonly used semi-supervised learning method for WSD. It works by iteratively classifying unlabeled examples and adding confidently classified examples into labeled dataset using a model learned from augmented labeled dataset in previous iteration. It can be found that the affinity information among unlabeled examples is not fully explored in this bootstrapping process. Bootstrapping 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 promising families of graph based semi-supervised learning algorithms are introduced, which can effectively combine unlabeled data with labeled in learning process by exploiting cluster structure in data . Label propagation algorithm is a graph 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 iteratively propagating label information from any vertex to nearby vertices through weighted edges, finally inferring the labels of unlabeled examples after this 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 unlabeled examples will have the same labels as labeled examples in the same cluster by propagating the label information of any example to nearby examples according to their proximity.

In Section 3.1 we describe Bootstrapping algorithms .Section 3.2 describe Graph based approach: label propagation algorithm

3.1 Bootstrapping Algorithms

To partly overcome the knowledge acquisition bottleneck, some methods have been devised for building sense classifiers when only a few annotated examples are available together with a high quantity of un-annotated data. These methods are often referred to as *bootstrapping* methods (Abney 2002, 2004). Among them, we can highlight *co-training* (Blum and Mitchell 1998), their derivatives (Collins and Singer 1999; Abney 2002, 2004), and *self-training* (Nigam and Ghani 2000). Briefly, co-training algorithms work by learning two complementary classifiers for the classification task trained on a small starting set of labeled examples, which are then used to annotate new unlabeled examples. From these new examples only the most confident predictions are added to the set of labeled examples. Each classifier is retrained with the additional training examples given by the other classifier, and the process repeats. The two complementary classifiers are constructed by considering two different *views* of the data (i.e., two different feature codifications), which must be conditionally independent given the class label. In several NLP tasks, co-training has provided moderate improvements with respect to not using additional unlabeled examples.

One important aspect of co-training consists on the use of different views to train different classifiers during the iterative process. While Blum and Mitchell (1998) stated the conditional independence of the views as a requirement, Abney (2002) shows that this requirement can be relaxed. Moreover, Clark et al. (2003) show that simply re-training on all newly labeled data can, in some cases, yield comparable results to agreement based co-training, with only a fraction of the computational cost.

Self-training starts with a set of labeled data, and builds a unique classifier (there are no different views of the data), which is then used on the unlabeled data. Only those examples with a confidence score over certain threshold are included in the new labeled set. The classifier is re-trained and the procedure repeated. Note the classifier uses its own predictions to teach itself. The procedure is also called self-teaching. Self-training has been applied to several natural language processing tasks. Yarowsky (1995) uses self-training for word sense disambiguation, e.g. deciding whether the word ‘plant’ means a living organism or a factory in a give context. Self-training is a wrapper algorithm, and is hard to analyze in general.

Mihalcea (2004) introduced a new bootstrapping scheme that combines co-training with majority voting, with the effect of smoothing the bootstrapping learning curves and improving the average performance. However, this approach assumes a comparable distribution of classes between both labeled and unlabeled data. At each iteration, the class distribution in the labeled data is maintained by keeping a constant ratio across classes between already labeled examples and newly added examples. This requires one to know a priori the distribution of sense classes in the unlabeled corpus, which seems unrealistic.

Pham et al. (2005) also experimented with a number of co-training variants on the Senseval-2 lexical sample and all-words tasks, including the ones in Mihalcea (2004). Although the original co-training algorithm did not provide any advantage over using only labeled examples, all the sophisticated co-training variants obtained significant improvements (taking Naïve Bayes as the base learning method). The best reported method was Spectral Graph Transduction Co-training.

3.1.1 Yarowsky Bootstrapping Algorithm

The Yarowsky algorithm (Yarowsky 1995) was, probably, one of the first and more successful applications of the bootstrapping approach to NLP tasks.

The Yarowsky algorithm is a simple iterative and incremental algorithm which eliminates the need for a large training set by relying on a relatively small number of instances of each sense for each lexeme of interest. These labeled instances are used as seeds to train an initial classifier using any of the supervised learning methods (decision lists, in this particular case). This initial classifier is then used to extract a larger training set from the remaining untagged corpus. Only those examples that are classified with a confidence above a certain threshold are kept as additional training examples for the next iteration. The algorithm repeats this retraining and re-labeling procedure until convergence (i.e., when no changes are observed from the previous iteration).

The key to this approach lies in its ability to create a larger training set from a small set of seeds Regarding the initial seed set, Yarowsky (1995a) discusses several alternatives to find them, ranging from fully automatic to manually supervised procedures. This initial labeling may have very low coverage (and, thus, low recall) but it is intended to have extremely high precision. As iterations proceed, the set of training examples tends to increase, while the pool of unlabeled examples shrinks. In terms of performance, recall improves with iterations, while precision tends to decrease slightly. Ideally, at convergence, most of the examples will be labeled with high confidence.

Discourse Properties: The Yarowsky Bootstrapping Algorithm Some well-known discourse properties are at the core of the learning process and allow the algorithm to generalize to label new examples with confidence.

We refer to: one-sense-per-collocation, language redundancy, and one-sense-per-discourse. First, the one-sense-per-collocation heuristic gives a good justification for using a decision list as the base learner, since a DL uses a single rule, based on a single contextual feature, to classify each new example. Second, we know that language is very redundant. This means that the sense of a concrete example is over determined by a set of multiple relevant contextual features (or collocations). Some of these collocations are shared among other examples of the same sense. These intersections allow the algorithm to learn to classify new examples, and, by transitivity, to increase recall with each iteration. This is the key point in the algorithm for achieving generalization. For instance, borrowing Yarowsky's (1995) original examples, a seed rule may establish that all the examples of the word plant in the collocation plant life should be labeled with the vegetal sense of plant (as opposed to its industrial sense). If we run a DL on the set of seed examples determined by this collocation, we may obtain many other relevant collocations for the same sense in the list of rules, for instance, "presence of the word animal in a ±10-word window". This rule would allow the classification of some new examples at the second iteration that were left unlabeled by the seed rule, for instance, the example contains a varied plant and animal life. Third, Yarowsky also applied the one-sense-per-discourse heuristic as a post-process at each iteration to uniformly label all the examples in the same discourse with the majority sense. This has a double effect. On the one hand, it allows the algorithm to extend the number of labeled examples, which, in turn, will provide new "bridge" collocations that cannot be captured directly from intersections among currently labeled examples. On the other hand, it allows the algorithm to correct some misclassified examples in a particular discourse.

Performance: Yarowsky's (1995) evaluation showed that, with a minimum set of annotated seed examples, this algorithm obtained comparable results to a fully supervised system (again, using DLs). The evaluation framework consisted of a small set of words limited to binary homographic sense distinctions.

Apart from simplicity, we would like to highlight another good property of the Yarowsky algorithm, which is the ability of recovering from initial misclassifications. The fact is that at each iteration all the examples are relabeled makes it possible that an initial wrong prediction for a concrete example may lower its

strength in subsequent iterations (due to the more informative training sets) until the confidence for that collocation falls below the threshold. In other words, we might say that language redundancy makes the Yarowsky algorithm self- correcting.

As a drawback, this bootstrapping approach has been theoretically poorly understood since its appearance in 1995. Recently, Abney (2004) performed some advances in this direction by analyzing a number of variants of the Yarowsky algorithm, showing that they optimize natural objective functions. Another criticism refers to real applicability, since Martínez and Agirre (2000) observed a far less predictive power of the one-sense per- discourse and one-sense-per-collocation heuristics when tested in a real domain with highly polysemous words.

3.1.2 Bilingual Bootstrapping Methods

A new method for word sense disambiguation, one that uses a machine learning technique called bilingual bootstrapping. In learning to disambiguate words to be translated, bilingual bootstrapping makes use of a small amount of classified data and a large amount of unclassified data in both the source and the target languages.

The data in the two languages should be from the same domain but are not required to be exactly in parallel. It repeatedly constructs classifiers in the two languages in parallel by repeating the following two steps: (1) Construct a classifier for each of the languages on the basis of classified data *in both languages*, and (2) use the constructed classifier for each language to classify unclassified data, which are then added to the classified data of the language. We can use classified data in both languages in step (1), because words in one language have translations in the other, and we can transform data from one language into the other. It boosts the performance of the classifiers by classifying unclassified data in the two languages and by exchanging information regarding classified data between the two languages.

The performance of bilingual bootstrapping been experimentally evaluated in word translation disambiguation, and all of their results indicate that bilingual bootstrapping consistently and significantly outperforms monolingual bootstrapping. The higher performance of bilingual bootstrapping can be attributed to its effective use of the asymmetric relationship between the ambiguous words in the two languages.

3.2 Label Propagation Algorithm

This algorithm 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 iteratively, finally inferring the labels of unlabeled examples after the propagation process converges.

3.2.1 Problem Setup

Let $X = \{x_i\}_{i=1}^{n}$ be 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 = $\{s_j\}_{j=1}^{c}$ denote 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 other 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 of x_g and similarity information among examples in *X*.

The cluster structure in *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: $W_{ij} = exp\left(-\frac{\|x_i - x_j\|^2}{\alpha^2}\right)$ if $i \neq j$ and $W_{ii} = 0$ $(1 \leq i, j \leq n)$, where α is used to control the weight W_{ij}.

3.2.2 Algorithm

In LP algorithm (Zhu and Ghahramani, 2002), label information of any vertex in a 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 closer examples, more likely they have similar labels (the global consistency assumption). In label propagation process, the soft label of each initial labeled example is clamped in each iteration to replenish 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 small 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 vertices, 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. 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.

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 $\overline{T}$ by $\overline{T_{ij}} = \frac{P_{ij}}{\sum_{k=1}^{n} P_{ik}}$.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 iteration index;
2. Propagate the label by $Y^{t+1} = TY^t$;
3. Clamp 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 $s_{\hat{j}}$,where $\hat{j} = argmax_j Y_{hj}$.

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

Performance The performance of LP been experimentally evaluated by taking reduced "interest" corpus (constructed by retaining four major senses) and complete "line" corpus as evaluation data.

Table 1: Accuracies from (Li and Li, 2004) and average accuracies of LP 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 Naïve 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	Accuracies from (Niu, Dong,Tan)		
	#labeled examples	LP_{cosine}	$LP_{JSMajor}$
interest	4×15=60	80.2±2.0%	79.8±2.0%
line	6×15=90	60.3±4.5%	59.4±3.9%

Table 1 summarizes the average accuracies of LP on the two corpora. It also lists the accuracies of monolingual bootstrapping algorithm (MB), bilingual bootstrapping algorithm (BB) on "interest" and "line" corpora. We can see that LP performs much better than MB-D and MB-B on both "interest" and "line" corpora, while the performance of LP is comparable to BB on these two corpora.

IV. Discussion

Primary focus of this paper is to discuss the semi-supervised algorithm for word sense disambiguation. Because use of small amount of labelled data and unlabeled data for semi supervised algorithms is gaining popularity in WSD filed.

Bootstrapping method (Yarowsky 1995) of semi-supervised learning was the first approach towards the word sense disambiguation.A great advantage of bootstrap is its simplicity. It is a straightforward way to derive estimates of standard errors and confidence intervals for complex estimators of complex parameters of the distribution, such as percentile points, proportions, odds ratio, and correlation coefficients. Moreover, it is an appropriate way to control and check the stability of the results. But bootstrapping is based on a local consistency assumption: examples close to labeled examples within same class.

Thus label propagation based semi-supervised learning algorithm for WSD came to the light, which fully realizes a global consistency assumption: similar examples should have similar labels.

In learning process, the labels of unlabeled examples are determined not only by nearby labeled examples, but also by nearby unlabeled examples.

Compared with semi-supervised WSD methods in the first and second categories, LP is the corpus based method and does not need external resources, including WordNet, bilingual lexicon, aligned parallel corpora.. It achieves better performance than SVM when only very few labeled examples are available, and its

performance is also better than monolingual bootstrapping and comparable to bilingual bootstrapping. By our investigation we can say LP is gaining more popularity in field of semi- supervised learning than Bootstrapping.

V. Conclusion

We described the Semi supervised learning methods and their performance in the context of word sense disambiguation. WSD is the important issue of natural language processing, which can be solved by using semi-supervised methods. These methods in comparison to other WSD approach are less time-consuming and inexpensive because they require only small amount of labelled training data and also use unlabeled data which is easily available.

References

[1] Dagan, I. & Itai A.. 1994. Word Sense Disambiguation Using A Second Language Monolingual Corpus. Computational Linguistics, Vol. 20(4), pp. 563- 596.
[2] Diab, M., & Resnik. P.. 2002. An Unsupervised Method for Word Sense Tagging Using Parallel Corpora. ACL-2002(pp. 255–262).
[3] 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.
[4] Ide, Nancy M. and Jean Veronis. 1998. Introduction to the special issue on word sense disambiguation: the state of the art. Computational Linguistics, 24(1), 1–40.
[5] Karov, Y. & Edelman, S.. 1998. Similarity-Based Word Sense Disambiguation. Computational Linguistics, 24(1): 41-59.
[6] Leacock, C., Miller, G.A. & Chodorow, M. 1998. Using Corpus Statistics and WordNet Relations for Sense Identification. Computational Linguistics, 24:1, 147–165.
[7] Lesk M.. 1986. Automated Word Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. Proceedings of the ACM SIGDOC Conference.
[8] Li, H. & Li, C.. 2004. Word Translation Disambiguation Using Bilingual bootstrapping. Computational Linguistics, 30(1), 1-22.
[9] Lin, J. 1991. Divergence Measures Based on the Shannon Entropy. IEEE Transactions on Information Theory, 37:1, 145–150.
[10] McCarthy, D., Koeling, R., Weeds, J., & Carroll, J.. 2004. Finding Predominan Word Senses in Untagged Text. ACL-2004.
[11] Mihalcea R.. 2004. Co-training and Self-training for Word Sense Disambiguation CoNLL-2004.
[12] Brown P., Stephen, D.P., Vincent, D.P., & Robert, Mercer.. 1991. Word Sense Disambiguation Using Statistical Methods. ACL-1991.
[13] Mo Yu, Shu Wang, Conghui Zhu, Tiejun Zhao .2011. Semi-supervised Learning for Word Sense Disambiguation using Parallel Corpora Laboratory of Natural Language Processing and Speech Harbin Institute of Technology Harbin, China
[14] Ng, H.T., Wang, B., & Chan, Y.S.. 2003. Exploiting Parallel Texts for Word Sense Disambiguation: An Empirical Study. ACL-2003, pp. 455-462.Park, S.B.,
[15] Schutze, H.. 1998. Automatic Word Sense Discrimination.Computational Linguistics, 24:1, 97–123.
[16] Yarowsky, D.. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. ACL-1995, pp. 189-196.
[17] Yarowsky, D.. 1992. Word Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. COLING-1992, pp. 454-460.
[18] Zhang, B.T., & Kim, Y.T.. 2000. Word Sense Disambiguation by Learning from Unlabeled Data. ACL-2000.
[19] Zheng-Yu Niu, Dong-Hong Ji & Chew Lim Tan. Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning
[20] Zhu, X. & Ghahramani, Z.. 2002. Learning from Labeled and Unlabeled Data with Label Propagation. CMU CALD tech report CMU-CALD-02-107.