# EXPLOITING TAGGED AND UNTAGGED CORPORA FOR WORD SENSE DISAMBIGUATION

ZHENGYU NIU
B.Eng., Tongji University
M.Eng., Tongji University

A THESIS SUBMITTED
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
SCHOOL OF COMPUTING
NATIONAL UNIVERSITY OF SINGAPORE

May 2006

# Acknowledgements

I would like to express my sincere appreciation to my supervisors, Dr. Dong Hong Ji at Institute for Infocomm Research and Prof. Chew Lim Tan at National University of Singapore for their continuous encouragement and guidance. It was, Dr. Ji and Prof. Tan, who guided me during my Ph.D. study at National University of Singapore. Their many helpful suggestions and comments have also been crucial to the completion of this thesis. Moreover, I would like to express my gratitude to the members of my dissertation committee: Prof. Hwee Tou Ng and Prof. Wee Sun Lee at National University of Singapore, who have been good enough to give this work a very serious review. Very special thanks are also due to Prof. Kim Teng Lua of National University of Singapore for his encouragement and guidance, particularly his supervision during my first year of Ph.D. study at National University of Singapore.

The research reported in this dissertation was conducted at Natural Language Synergy Lab, Media Division, Institute for Infocomm Research. I would like to express my sincere appreciation to my colleagues at Natural Language Synergy Lab, Mr. Ling Peng Yang, Mr. Yu Nie, Mr. Xiao Feng Yang, Ms. Jin Xiu Chen, Mr. Jie Zhang, Ms. Juan Xiao, Ms. Dan Shen, Dr. Li Tang, Dr. Min Zhang, Dr. Guo Dong Zhou, Dr. Jian Su, Ms. Ai Ti Aw, my friends at National University of Singapore, Mr. Xi Ma, Mr. Xing Lei Zhu, Mr. Zhi Cheng Zhou, Mr. Shui Ming Ye, Ms. Rong Zhang, Ms. Rui Li, Mr. Xi Shao, Mr. Yan Tao Zheng, Mr. Jin Jun Wang, Ms. Yong Kwan Lim, and my friends in Singapore, Dr. Kai Chen, Dr. Yang Xiao, Mr. Liang Huang, Mr. Xiao Jun Fu. Without their continuous encouragement and support, I would not have been able to complete this work. I owe a great many thanks to many people who were kind enough to help me over the course of this work. I would like to express here my great appreciation to all of them. Finally, I also would like to express a deep debt of gratitude to my parents for their every concern and support.

# Contents

# Summary

In traditional supervised methods to sense disambiguation, one uses only sense tagged corpora to train sense taggers. Sense tagged examples are often difficult, expensive, or time consuming to obtain, as they require the efforts of experienced human annotators. Meanwhile untagged corpora may be relatively easy to collect, but there have been few ways to use them. Unsupervised sense disambiguation methods address this problem by using only a large amount of untagged corpora to discriminate the instances of an ambiguous word.

However the sense clustering result by unsupervised methods cannot be directly used in many natural language processing tasks since there is no sense tag for each instance in clusters. Considering both the availability of a large amount of untagged corpora and the direct use of word senses, semi-supervised learning has received great attention recently. Semi-supervised sense disambiguation methods use a large amount of untagged corpora, together with the sense tagged corpus, to build better sense taggers.

If there are no tagged examples for a sense (e.g., a domain specific sense) in the sense tagged corpus and there is a large amount of untagged corpora that contain instances for both general senses and the missed sense, then a sense tagger built on the incomplete sense tagged corpus will mis-tag the instances of the missed sense. It is a problem encountered by traditional supervised or semi-supervised sense disambiguation methods. Partially supervised learning addresses this problem by identifying a set of reliable sense tagged examples from the untagged corpus for the missed sense, and then building a sense tagger with the learned sense tagged data.

We investigate a series of novel machine learning approaches on benchmark corpora for sense disambiguation and empirically compare them with other related state of the art sense disambiguation methods. They address the following questions: How to automatically estimate the number of senses (or sense number, model order) of an ambiguous word from an untagged corpus? (Minimum Description Length criterion); How to use untagged corpora to build a better sense tagger? (label propagation); How to perform sense disambiguation with an incomplete sense tagged corpus? (partially supervised learning). This thesis includes an extensive literature review for sense disambiguation and other related work.

# List of Tables

# List of Figures

# Chapter 1

# Introduction

In this chapter, we present an overview of word sense disambiguation (WSD), including the motivation and definition of WSD. Then we provide a review on advances in automatic sense disambiguation methods. Finally we present the motivation and objective of our work on sense disambiguation.

The automatic methods for WSD include knowledge based methods, hybrid methods, and corpus based methods (or statistical methods).

With the availability of large scale lexical resources such as dictionaries and thesauri, knowledge based methods were proposed to automatically extract knowledge from these sources. But these lexical resources are not adequate for WSD, since they provide detailed information only at the lexical level, lacking pragmatic information for sense determination. Therefore, with the availability of very large corpora, corpora have become a primary source of information for WSD.

Some hybrid methods were proposed to extract information from large untagged corpora as supplement to the information in lexical resources for sense disambiguation.

Corpus based methods include supervised sense disambiguation methods, unsupervised sense disambiguation methods, and semi-supervised sense disambiguation methods (or weakly supervised sense disambiguation methods). Unsupervised sense disambiguation methods do not require sense tagged corpus and pre-defined sense inventory, which have been investigated in previous study. But previous methods usually require the specification of sense number by users. For solving this problem, we present an unsupervised sense discrimination algorithm to induce senses of a target word by grouping its occurrences into a "natural" number of clusters based on the similarity of their contexts.

However, the results from unsupervised methods cannot be directly used in many natural language processing (NLP) tasks since there is no sense tag attached to each instance in clusters. Considering both the availability of a large amount of untagged corpora and direct usage of word senses, semi-supervised sense disambiguation methods such as bootstrapping, have received great attention recently. These semi-supervised methods are based on a local consistency assumption: examples near to the same labeled example are likely to have the same label, which is also the assumption underlying many supervised learning algorithms, such as kNN. Furthermore, it can be found that the affinity information among unlabeled examples is not fully explored in the bootstrapping process. In other words, these algorithms do not use the similarity of unlabeled data to smooth their labels. Recently, a promising

semi-supervised learning method, the label propagation algorithm [164], has been introduced in machine learning community, which represents labeled and unlabeled examples and their distances as the nodes and the weights of edges of a graph, and tries to obtain a labeling function to satisfy two constraints: 1) it should be fixed on the labeled nodes, 2) it should be smooth on the whole graph. Here we would like to investigate this label propagation based semi-supervised learning algorithm for sense disambiguation.

Supervised and semi-supervised sense disambiguation methods will mis-tag instances of a target word if the senses of these instances are not defined in sense inventories or there are no tagged instances for these senses in training data. We propose an automatic method, a partially supervised sense disambiguation algorithm, to avoid the misclassification of the instances with undefined senses by discovering new senses from mixed data (tagged corpus+untagged corpus). This algorithm can obtain a natural partition of mixed data by maximizing a stability criterion defined on classification results from an extended label propagation algorithm over all the possible values of sense number (or the number of senses, or model order).

Next we provide the motivation and definition of automatic word sense disambiguation in section 1.1. Then section 1.2 provides a review on the development of automatic sense disambiguation methods. Section 1.3 presents the motivation and objective of our work.

## 1.1 Overview of Word Sense Disambiguation

In many natural languages, most of the words have many possible meanings. When using a computer program to automatically process a natural language, the sense ambiguity problem arises, since the computer program has no basis for knowing which sense is appropriate for a word in a given context. Therefore automatic word sense disambiguation is an important intermediate task for language understanding systems such as machine translation [147], information retrieval [123, 125], and speech processing [133, 156].

Word sense disambiguation can be defined as associating a given word in a text or discourse with a definition or meaning. Many automatic methods have been proposed to deal with this sense disambiguation problem, including knowledge based methods, hybrid methods, and corpus based methods. In next section, we provide a review on the development of automatic sense disambiguation methods.

## 1.2 Previous Work on Word Sense Disambiguation

### 1.2.1 Knowledge Based Sense Disambiguation

In the early 1960's, the problem of sense disambiguation in language understanding systems was usually handled by rule based methods [4, 27, 54, 83, 85, 150]. They involved the use of detailed knowledge of syntax and semantics, which required much human effort and time to generate. The difficulty of hand-crafted knowledge sources restricts these rule based methods to "toy" implementations handling only a tiny fraction of the language.

With the availability of large scale lexical resources, the work on WSD reached a turning point in the 1980s. Knowledge based methods were proposed to automatically extract

knowledge from manually constructed lexical resources for sense disambiguation [55, 71, 76, 82, 84, 116, 129, 143, 151, 154].

Lesk (1986) presented an automatic method to perform disambiguation by selecting the sense of a target word whose definition contained the greatest number of word overlaps with the neighboring words in its context. This method achieved 50-70% correct disambiguation, using a relatively fine set of sense distinctions such as those found in a typical learner's dictionary. Lesk's method is sensitive to the exact wording of each definition: the presence or absence of a given word can radically alter the results. However, Lesk's method has served as the basis for most Machine Readable Dictionary (MRD) based disambiguation work that has followed.

Wilks et al. (1990) attempted to improve the knowledge associated with each sense by calculating the frequency of co-occurrence for the words in definition texts, from which they derived several measures of the degree of relatedness among words. This metric was then used with the help of a vector method that related each word and its context. In experiments on a single word (bank), the method achieved 45% accuracy on sense identification, and 90% accuracy on homograph identification.

Veronis and Ide (1990) extended Lesk's method by automatically building very large neural networks (VLNNs) from definition texts in machine-readable dictionaries, and demonstrated the use of these networks for word sense disambiguation. In the VLNNS, each word was linked to its senses, which were themselves linked to the words in their definitions, which were in turn linked to their senses, etc.. They showed an application of this method to sense disambiguation on the word "pen". They concluded that their method is more robust than the Lesk's strategy, since it does not rely on the presence or absence of a particular word or words and can filter out some degree of "noise" (such as inclusion of some wrong lemmas due to the lack of information about part-of-speech or occasional activation of misleading homographs).

Another resource for sense disambiguation is thesaurus, which can provide information about relationships among words, most notably synonymy. Roget's International Thesaurus, which was put into machine-tractable form in the 1950's [82], supplies an explicit concept hierarchy consisting of up to eight increasingly refined levels. It has been used in a variety of applications including machine translation, information retrieval, and content analysis.

Masterman (1957) applied Roget's International Thesaurus to the problem of WSD: in an attempt to translate Virgil's Georgics by machine, she looked up, for each Latin word stem, the translation in a Latin-English dictionary and then looked up this word in the word-to-head index of Roget's. In this way each Latin word stem was associated with a list of Roget head numbers associated with its English equivalents. The numbers for words appearing in the same sentence were then examined for overlaps. Finally, English words appearing under the multiply-occurring head categories were chosen for the translation.

In the mid-1980s, several efforts began to construct large scale knowledge bases by hand (e.g., WordNet [91]). WordNet is at present the best known and the most utilized resource for word sense disambiguation in English, since it provides the broadest set of lexical information in a single resource, and it is freely and widely available. WordNet combines the features of many of the other resources commonly exploited in disambiguation work: it includes definitions for individual senses of words within it, as in a dictionary; it defines "synsets" of synonymous words representing a single lexical concept, and organizes them into a conceptual

hierarchy, like a thesaurus; and it includes other links among words according to several semantic relations, including hyponymy/ hypernymy, antonymy, meronymy, etc..

Resnik (1995) explored a measure of semantic similarity for words in the WordNet hierarchy. He computed the shared "information content" of words, which was a measure of the specificity of the concept that subsumed the words in the WordNet IS-A hierarchy–the more specific the concept that subsumed two or more words, the more semantically related they were assumed to be. Resnik contrasted his method of computing similarity to those which compute path length, arguing that the links in the WordNet taxonomy do not represent uniform distances. Resnik's method, applied using WordNet's fine-grained sense distinctions and measured against the performance of human judges, approached human accuracy.

Mihalcea (2005) presented a graph based algorithm to solve the all-words WSD problem, which exploited the dependencies between senses of different words. The author's graph based sequence labeling algorithm consisted of three steps: graph construction, scoring vertices in graph, and label assignment for each word. In graph construction phase, all possible senses of all words in an input sentence were represented as vertices. The vertices within a maximum allowable distance were connected by edges, and each edge was associated with a weight. Weights of each edge were computed using Lesk-like method: normalized number of common tokens between definitions of two senses. Next, scores were assigned to vertices using a graph based ranking method, the PageRank algorithm. Finally, the most likely set of labels was determined by identifying for each word the label that had the highest score. This algorithm was evaluated on SENSEVAL-2 and SENSEVAL-3 all-words task data set. It outperforms random baseline, Lesk method, McCarthy's method, and the method by R.Mihalcea (2004c). The algorithm differs from that in Mihalcea (2004c) by using knowledge-lean method to calculate the similarity between vertices without the use of semantic network in WordNet.

### 1.2.2 Hybrid Methods for Sense Disambiguation

With the availability of large scale raw corpora, some sense disambiguation methods [76, 84, 129, 154] try to extract the information in raw corpora as supplement to the information in lexical resources.

Yarowsky (1992) addressed the problem of knowledge acquisition bottleneck by tagging each target word with the semantic categories in Roget's thesaurus to automatically generate a non-perfect sense-tagged corpus. He reported 92% accuracy on a mean 3-way sense distinction. Yarowsky noted that his method is best for extracting topical information, which is in turn most successful for disambiguating nouns.

Lin (1997) presented an algorithm that uses WordNet to disambiguate different words. The algorithm does not require a sense-tagged corpus and exploits the fact that two different words are likely to have similar meanings if they occur in identical local contexts. Finally Lin evaluated this algorithm on polysemous nouns in SemCor corpus and empirically compared it with a baseline which always selected the first sense in WordNet. Lin's algorithm performed slightly worse than the baseline when the strictest correctness criterion ($sim(s_{answer}, s_{key}) = 1$) was used. However, when the condition ($sim(s_{answer}, s_{key}) > 0$ or $\geq 0.27$) was relaxed, its performance gain was much larger than the baseline. This means that when the algorithm makes mistakes, the mistakes tend to be close to the correct answer. $sim(s_{answer}, s_{key}) = 1$ is

true only when $s_{answer} = s_{key}$. The most relaxed interpretation $sim(s_{answer}, s_{key}) > 0$ is true if $s_{answer}$ and $s_{key}$ are the descendants of the same top-level concepts in WordNet (e.g., entity, group, location, etc.). A compromise between these two criteria is $sim(s_{answer}, s_{key}) \geq 0.27$, where 0.27 is the average similarity of 50,000 randomly generated pairs (w, w') in which w and w' belong to the same Roget's category.

McCarthy et al. (2004) proposed a method that used raw corpus to automatically find a predominant sense for nouns in WordNet. They used an automatically acquired thesaurus and a WordNet Similarity measure. The automatically acquired predominant senses were evaluated against the hand-tagged resources SemCor and the SENSEVAL-2 English all-words task giving them a WSD precision of 64% on an all-nouns task. This was just 5% lower than results using the first sense in the manually labeled SemCor, and they obtained 67% precision on polysemous nouns that were not in SemCor.

Seo et al. (2004) described a statistical model to determine preferred sense among WordNet relatives of an ambiguous word in a given context of its occurrence by the use of WordNet and co-occurrence frequency (calculated from untagged corpora) between candidate relatives and each word in the context. Experiment results on the data of English lexical sample (ELS) task of SENSEVAL-2 indicated that their method achieved 45.48% precision and recall, which slightly outperforms the best automatic unsupervised system in ELS task of SENSEVAL-2.

### 1.2.3 Corpus Based Sense Disambiguation

In the 1980's the interest in corpus linguistics was revived. Advances in technology enabled the creation and storage of corpora larger than what was possible previously. Furthermore, the availability of these corpora enabled the application of statistical models to extract sense disambiguation information from corpora for WSD. Corpus based Methods include supervised methods, semi-supervised methods, and unsupervised methods.

**Supervised Sense Disambiguation**

Supervised methods usually rely on the information from previous sense tagged corpora to determine the senses of words in unseen texts [12, 67, 48, 20, 105, 70, 98, 95, 109, 152, 157].

Black (1988) developed a model based on decision trees using a corpus of 22 million tokens, after manually sense-tagging approximately 2000 concordance lines for five test words. Since then, supervised learning from sense-tagged corpora has been used by several researchers: [67, 48, 20, 105, 70, 98, 95, 109, 152, 157].

Pedersen (2000) presented a corpus-based approach to word sense disambiguation that built an ensemble of Naive Bayesian classifiers, each of which was based on lexical features that represented co-occurring words in varying sized windows of context. Experimental results on "line" and "interest" corpora showed that such an ensemble achieved higher accuracy than previous methods, e.g. kNN [98], probabilistic model [20], and Naive Bayesian classifier [67, 95].

In ELS task of SENSEVAL-2, the top three systems are JHU [158], SMUls, and KUNLP. JHU employed an ensemble of three classifiers (cosine based vector models, Bayesian models, and decision list) with various knowledge sources such as surrounding words, local collocations, syntactic relations, and morphological information. SMUls used a k-nearest neighbor algorithm with features such as keywords, collocations, POS, and name entities. KUNLP

used Classification Information Model, an entropy-based learning algorithm, with local, topical, and bigram contexts and their POS tags.

Lee and Ng (2002) empirically examined the interaction of different classifiers (SVM, Adaboost, Naive Bayes, decision list) with various features (Part-of-Speech of neighboring words, unordered words in surrounding context, local collocation, syntactic relation) and concluded that an SVM using all the available features without feature selection achieved the highest accuracy on official data in ELS task of SENSEVAL-1 and 2, and outperforms previous top systems in SENSEVAL-1 and 2.

In ELS task of SENSEVAL-3 [88], the top three systems are htsa3, IRST-Kernels, and nusels. htsa3 used a Naive Bayes system, with correction of the a-priori frequencies, dividing the output confidence of the senses by $frequency_\alpha$ ($\alpha = 0.2$). But how to determine the value of $\alpha$ is still an open problem. IRST-Kernels used an SVM classifier with paradigmatic and syntagmatic information and unsupervised term proximity (LSA) on BNC. nusels used a combination of various knowledge sources (part-of-speech of neighboring words, words in context, local collocations, syntactic relations), in an SVM classifier. We can see that the second and third top performing systems used SVM as a classifier, while several of other top performing systems were based on combinations of multiple classifiers.

Based on the results in previous study, we can see that SVM and ensemble method using local and topical features are state of the art techniques for WSD.

However, despite the availability of increasingly large corpora and the success of supervised sense disambiguation methods, the difficulties of manually sense-tagging a training corpus impedes the acquisition of lexical knowledge from corpora.

Many semi-supervised methods have been proposed to automatically augment sense-tagged corpora or use untagged corpora to improve the performance of sense tagger trained from small tagged corpora [18, 29, 38, 51, 60, 74, 87, 99, 107, 155], which are reviewed later.

Another problem encountered by supervised WSD is domain dependence: a system trained on corpora from one domain (e.g., finance), will show a decrease in performance when applied to a different domain (e.g., sports). Escudero et al. (2000) conducted a set of comparative experiments cross different corpora. They concluded that the domain dependence of WSD systems seems very strong and suggested that some kind of adaptation or tuning is required for cross-corpus application. Motivated by the observation that different sense distributions across domains have an important effect on WSD accuracy [42, 1], Chan and Ng (2005) used two distribution estimation algorithms to provide estimates of the sense distribution in a new data set. The results on the nouns of the SENSEVAL-2 English lexical sample task showed that their methods are effective in improving the accuracy of sense disambiguation on different domains. Gliozzo et al. (2004) extended and grounded the modeling of domains and the exploitation of WordNet Domains, an extension of WordNet in which each synset is labeled with domain information. They proposed a novel unsupervised probabilistic method for the critical step of estimating domain relevance for contexts, and suggested utilizing it within unsupervised Domain Driven Disambiguation for word senses, as well as within a traditional supervised approach.

**Semi-Supervised Sense Disambiguation**

Supervised sense disambiguation methods require a lot of manually sense-tagged corpus that is difficult to acquire, while the results from unsupervised methods cannot be directly used in many NLP tasks since there is no sense tag attached to each instance in clusters.

Considering both the availability of a large amount of untagged corpora and direct usage of word senses, many efforts have been devoted to semi-supervised methods recently [18, 29, 38, 51, 60, 74, 87, 99, 107, 111, 155].

Semi-supervised sense disambiguation methods are characterized in terms of exploiting untagged corpora in the learning procedure with predefined sense inventories for ambiguous words.

Some methods were proposed to exploit bilingual resources, e.g., aligned parallel corpora, untagged monolingual corpora in two languages. The intuition behind these methods is that if different senses of an ambiguous word in the source language are translated into different words in the target language, then translated words in the target language can serve as tags of the senses of this ambiguous word.

Brown et al. (1991) employed a flip-flop algorithm to derive sense disambiguation questions in the source language from a large aligned parallel corpus. Then questions about the contexts of instances of an ambiguous word were used for sense disambiguation of this word. The incorporation of this disambiguation method improved their statistical machine translation system. The aligned parallel corpus required by their method was a result of manual translation. Gale et al. (1992) and Ng et al. (2003) also exploited aligned parallel copora to generate large sense-tagged training data for WSD.

Dagan and Itai (1994) proposed a sense disambiguation method that requires only a bilingual lexicon and a monolingual corpus, which may avoid the requirement of aligned bilingual corpora in the above sense disambiguation methods. Their algorithm disambiguated senses of words in the source language by three steps: (1) identify syntactic relations between words in the source language; (2) map the alternative interpretations of these relations to the target language using a machine translation system; (3) select the preferred senses according to statistics on lexical relations and lexical constraints in the target language.

Different from the work in Dagan and Itai (1994), Diab and Resnik (2002) exploited a knowledge based method to disambiguate ambiguous words in the source language. Firstly, they translated sentences with an ambiguous word into the target language. Then the information from WordNet was used to disambiguate a group of translations in the target language that corresponded to the same ambiguous word in the source language. Finally, they projected sense tags between the two languages to automatically generate aligned parallel sense-tagged corpora, which can be used as the source of training data for WSD.

Li and Li (2004) presented a bilingual bootstrapping algorithm, which can boost the performance of sense classifiers in two languages by repeatedly tagging the text of words related to the same sense in both languages and exchanging the information regarding the tagged text of the same sense between the two languages. Experiment results on benchmark corpora showed that untagged data in the second langauge does help the sense disambiguation in the first language, which leads to the better performance of bilingual bootstrapping in comparison with monolingual bootstrapping.

Another research line is to automatically generate monolingual sense tagged corpus without reference to the second language corpora. Bootstrapping (or self-training) is such a general scheme for minimizing the requirement of manually tagged corpus, which was proposed for sense disambiguation [51]. Bootstrapping method augments an initial set of manually sense-tagged data by iteratively training a base classifier on tagged data, using the resulting classifier to disambiguate additional untagged data, and adding the most confidently tagged

examples to tagged data till a stopping criterion is satisfied.

Hearst's bootstrapping method was improved by Yarowsky (1995) in two aspects: (a) manually identify collocations for word senses to generate initial labeled data; (b) exploit a redundant view (one sense per discourse property) to filter or augment sense tagged examples in the bootstrapping process.

Some efforts were devoted to improve the base classifier in the bootstrapping process: Park et al. (2000) used committee learning algorithm as the base classifier, while Mihalcea (2004a) introduced a combination of majority voting with bootstrapping or co-training.

Karov and Edelman (1998) proposed another approach to automatically augment sense tagged corpus for WSD. It combined untagged sentences and sense related sentences of the same ambiguous word from a lexicon for learning contextual word similarity and sentence similarity. Additional sense tagged corpus can be obtained by assigning each untagged sentence with the sense of its most similar sense related sentence by the use of sentence similarity.

Data from the web demonstrates enormous potential for NLP tasks. Mihalcea and Moldovan (1999) did some work on using web data to obtain sense tagged corpus. They used the information from WordNet to formulate queries consisting of synonyms or definitions of word senses, and obtained additional training data for word senses from Internet using existing search engines.

Recently, Pham et al. (2005) described an application of four semi-supervised learning algorithms for WSD, including basic co-training, smoothed co-training, spectral graph transduction (SGT), and a variant of SGT (SGT+co-training). Their results showed that the variant of SGT achieves the best performance, compared to the other three semi-supervised algorithms.

**Unsupervised Sense Disambiguation**

Unsupervised methods discriminate senses for an ambiguous word by grouping its occurrences into a specified number of clusters based on the similarity of their contexts without the need of sense definition and sense tagged corpus.

Schütze (1998) presented a context group discrimination algorithm for unsupervised sense disambiguation. Firstly, their algorithm selected important contextual words using $\chi_2$ or local frequency criterion. With the $\chi_2$ based criterion, those contextual words whose occurrence depended on whether the ambiguous word occurred were chosen as features. When using local frequency criterion, their algorithm selected top n most frequent contextual words as features. Then each context of occurrences of the target word was represented by second order co-occurrence based context vector. Singular value decomposition (SVD) was conducted to reduce the dimensionality of context vectors. Then the reduced context vectors were grouped into a pre-defined number of clusters whose centroids corresponded to senses of the target word.

Pedersen and Bruce (1997) conducted an experimental comparison of three clustering algorithms for word sense discrimination. Their feature sets included morphology of a target word, part of speech of contextual words, absence or presence of particular contextual words, and collocation of frequent words. Then occurrences of a target word were grouped into a pre-defined number of clusters based on the similarity of feature vectors. Similar with many other algorithms, their algorithm also required the cluster number to be provided.

Fukumoto and Suzuki (1999) proposed a term weight learning algorithm for verb sense

disambiguation, which can automatically extract nouns co-occurring with verbs and identify the number of senses of an ambiguous verb. The weakness of their method is to assume that nouns co-occurring with verbs are disambiguated in advance and the number of senses of the target verb is no less than two.

Chen and Palmer (2004) discussed an application of the Expectation-Maximization (EM) clustering algorithm to the task of Chinese verb sense discrimination. Their model utilized rich linguistic features that captured predicate-argument structure information of a target verb. The number of clusters was required to be provided in their algorithm, which was set to be identical with the ground-truth value of sense number of the target verb.

Word clustering may be considered as closely related work to sense discrimination. It treats a word sense as a set of synonyms like synset in WordNet. Many methods are proposed for clustering related words using information acquired from raw texts [19, 30, 39, 144] or parsed/chunked corpora [21, 53, 77, 106, 110].

Brown et al. (1992) proposed a class based n-gram model to address the problem of predicting a word from previous words in a sample of text. It worked by grouping words into classes of similar words, so that one can base the estimate of a word pair's probability on the averaged co-occurrence probability of the classes to which the two words belong.

Dagan et al. (1997) described a similarity-based estimation method to address the problem of estimating the probability of unseen word pairs in training data. When encountering an unseen word pair $< w_1, w_2 >$, estimates for $w_1$'s most similar words $\bar{w}_1$ were combined as the probability estimate for this word pair by weighting the evidence provided by $\bar{w}_1$ based on the similarity between $w_1$ and $\bar{w}_1$.

Dorow and Widdows (2003) proposed to represent a target noun word, its neighbors and their relationships using a graph in which each node denoted a noun and two nodes had an edge between them if they co-occurred with more than a given number of times. Then senses of the target word were iteratively learned by clustering the local graph of similar words around the target word. Their algorithm required a threshold as input, which controlled the number of senses.

Veronis (2004) developed an algorithm called HyperLex that is capable of automatically determining word uses in an unseen text without recourse to a dictionary. This algorithm made use of the specific properties of word co-occurrence graphs, which were shown as having "small world" properties. Unlike earlier dictionary-free methods based on word vectors, it can isolate highly infrequent uses (as rare as 1% of all occurrences) by detecting "hubs" and high-density components in the co-occurrence graphs. This algorithm was applied to information retrieval on the Web, using a set of highly ambiguous test words. Experiment results showed that it only omitted a very small number of relevant uses. In addition, HyperLex offered automatic tagging of word uses in context with excellent precision.

Hindle (1990) described a method of determining the similarity of nouns on the basis of a metric derived from the distribution of subject, verb and object in a large text corpus. The resulting quasi-semantic classification of nouns demonstrated the plausibility of the distributional hypothesis, and had potential applications to a variety of tasks, including automatic indexing, resolving nominal compounds, and determining the scope of modification.

Pereira et al. (1993) described and evaluated a method for clustering words according to their distribution in particular syntactic contexts. Words were represented by the relative frequency distributions of contexts in which they appeared, and relative entropy between

those distributions was used as the similarity measure for clustering.

Lin (1998) presented a method for automatic construction of thesaurus by clustering related words using a word similarity measure based on the distributional syntactic pattern of words.

The approach proposed by Caraballo (1999) can find both the sets of related words, and then the relationships between those sets. The sets of words were found using syntactic clues, particularly conjunctions of noun phrases as well as appositives.

Pantel and Lin (2002)'s method initially discovered tight clusters called committees by grouping top $n$ words similar with a target word using average link clustering. Then the target word was assigned to committees if the similarity between them was above a given threshold. Each committee that the target word belonged to was interpreted as one of its senses.

## 1.3 Motivation and Objective of This Work

### 1.3.1 Word Sense Discrimination with Feature Selection and Order Identification Capabilities

Sense disambiguation is essential for many language understanding systems such as information retrieval, speech processing, and text processing [56]. Many methods have been proposed to deal with this problem, including knowledge based methods, hybrid methods, and corpus based methods (e.g., supervised learning algorithms, semi-supervised learning algorithms, and unsupervised learning algorithms).

Supervised sense disambiguation methods usually rely on the information from previous sense tagged corpus to determine the senses of words in an unseen text. They require a lot of sense tagged corpora, and heavily depend on manually compiled lexical resources as sense inventories. However, these lexical resources often miss domain specific word senses, and even many new words are not included inside. Learning word senses from untagged corpora may help us dispense with the need for an outside knowledge source for defining senses by only discriminating senses of words.

Word sense can be represented as a group of similar contexts of a target word. The context group discrimination ($CGD$) algorithm [126] adopts this strategy.

Some observations can be made about the feature selection and clustering procedure in the $CGD$ method. One observation is that their feature selection uses only first order information although the second order co-occurrence data is available. The other observation is about their clustering procedure. Their method can capture both coarse-gained and fine-grained sense distinction as the predefined cluster number varies. But from a point of statistical view, there should exist a partitioning of data at which the most reliable, "natural" sense clusters appear.

In this work, we follow the second order representation method for contexts of a target word, since it is supposed to be less sparse and more robust than the first order information [126]. A cluster validation based unsupervised feature wrapper is introduced to remove noises in the contextual word set, which works by measuring the consistency between cluster structures estimated from disjoint data subsets in selected feature space. It is based on

the assumption that if selected feature subset is important and complete, cluster structure estimated from data subset in this feature space should be stable and robust against random sampling. After determination of important contextual words, a Gaussian mixture model (GMM) based clustering algorithm [16] is used to estimate cluster structure and cluster number by minimizing Minimum Description Length (MDL) criterion [119].

The aim of this work is to

(1) describe a GMM+MDL based sense discrimination algorithm;

(2) evaluate this algorithm on benchmark data (the "hard", "interest", "line", and "serve" corpora) and empirically compare it with a state of the art method, *CGD* algorithm, for sense discrimination.

### 1.3.2 Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning

Semi-supervised learning methods have received great attention recently for sense disambiguation, since they can use both labeled and unlabeled data, and they can achieve better performance than supervised methods in most of cases.

Semi-supervised methods for WSD are characterized in terms of exploiting unlabeled data in the learning procedure with the requirement of predefined sense inventories for target words. As a commonly used semi-supervised learning scheme for WSD, bootstrapping [51] works by iteratively classifying unlabeled examples and adding confidently classified examples into labeled data using a model learned from augmented labeled data in the previous iteration. We can see that it is based on a local consistency assumption: examples near to the same labeled example are likely to have the same label, which is also the assumption underlying many supervised learning algorithms, such as kNN. Furthermore, the affinity among unlabeled examples is not fully explored in this bootstrapping process.

Recently, a promising semi-supervised learning method, the label propagation algorithm (LP) [164], has been introduced in the machine learning community, which represents labeled/unlabeled examples and their distances as the nodes and the weights of edges of a graph, and tries to obtain a labeling function to satisfy two constraints: 1) it should be fixed on the labeled nodes, 2) it should be smooth on the whole graph. Compared with bootstrapping, LP can utilize the cluster structure in unlabeled examples by smoothing the labeling function on the whole graph.

This work investigates this graph based method for WSD, which can fully exploit the cluster structure in unlabeled data in classification process. Specifically, the aim of this work is to

(1) evaluate the LP algorithm for WSD on benchmark data (the "interest" corpus, the "line" corpus, the SENSEVAL-2 corpus, and the SENSEVAL-3 corpus);

(2) empirically compare the LP algorithm with other methods for WSD, e.g., SVM, bootstrapping, co-training and their variants with majority voting.

### 1.3.3 Partially Supervised Sense Disambiguation by Learning Sense Number from Tagged and Untagged Corpora

Many algorithms have been proposed to deal with the sense disambiguation problem when given definition for each possible sense of a target word or tagged corpus with instances of all possible senses, e.g., supervised sense disambiguation [67], and semi-supervised sense disambiguation [155].

Supervised methods usually rely on the information from previous sense tagged corpora to determine the senses of words in unseen texts. Semi-supervised methods for WSD are characterized in terms of exploiting unlabeled data in the learning procedure with the need of predefined sense inventories for target words. The information for semi-supervised sense disambiguation is usually obtained from bilingual corpora (e.g. parallel corpora or untagged monolingual corpora in two languages) [18, 29, 74], or sense-tagged seed examples [155].

Some observations can be made on previous supervised and semi-supervised methods. They always rely on hand-crafted lexicons as sense inventories. But these resources may miss domain-specific senses, which leads to incomplete sense tagged corpus [1]. Therefore, sense taggers trained on the incomplete tagged corpus will misclassify the instances with senses undefined in sense inventories. For example, one performs WSD in information technology related texts using WordNet [2] as sense inventory. When disambiguating the word "boot" in the phrase "boot sector", the sense tagger will assign this instance with one of the senses of word "boot" listed in WordNet. But the correct sense "loading operating system into memory" is not included in WordNet. Therefore, this instance will be associated with an incorrect sense.

Unsupervised sense discrimination methods do not rely on predefined sense inventory, which may be used to solve this problem. But they cannot use the labeling information in sense tagged corpora. Moreover, the results from unsupervised methods cannot be directly used in many NLP tasks since generally there is no sense tag attached to each instance.

So, in this work, we would like to study the problem of partially supervised sense disambiguation with incomplete sense tagged corpus. Specifically, given incomplete sense-tagged examples and a large amount of untagged examples for a target word, we are interested in (1) labeling the instances of a target word in untagged corpus with sense tags occurring in the tagged corpus; (2) finding undefined senses of the target word from the untagged corpus if they occur in the untagged corpus, which will be represented by instances from the untagged corpus.

We propose an automatic method to estimate the sense number of a target word in mixed data (tagged corpus+untagged corpus) by maximizing a stability criterion defined on classification results over all the possible values of sense number. At the same time, we can obtain a classification result on the mixed data. If the estimated sense number in the mixed data is equal to the sense number of the target word in tagged corpus, then there is no new sense in untagged corpus. Otherwise new senses will be represented by groups in which there is no instance from the tagged corpus. The stability criterion assesses the agreement between classification results on full mixed data and sampled mixed data. A

[1] "incomplete sense tagged corpus" means that the sense tagged corpus does not include the instances of some senses for a target word, while these senses may occur in unseen texts.

[2] Online version of WordNet is available at http://wordnet.princeton.edu/cgi-bin/webwn2.0

partially supervised learning algorithm is used to classify mixed data into a given number of classes before stability evaluation. The class number for partially supervised learning is no less than the class number in the tagged corpus.

This sense number estimation process is necessary since it is usually unknown whether there is any new sense in the untagged corpus. This partially supervised sense disambiguation method may help us to conduct sense disambiguation when not all the senses are given in training data.

The aim of this work is to

(1) present a partially supervised sense disambiguation algorithm;

(2) evaluate it on benchmark data (the SENSEVAL-3 corpus) and empirically compare it with other related algorithms, e.g., a one-class partially supervised classification algorithm [80], and a clustering based partially supervised sense disambiguation algorithm.

Partially supervised sense disambiguation in untagged corpora helps sense disambiguation systems to avoid misclassification of the instances with undefined senses. Another possible application of this partially supervised sense disambiguation algorithm is to help enrich manually compiled lexicons by learning new senses from untagged corpora.

### 1.3.4 Thesis Structure

Next chapter (Chapter 2) provides a review on related work, e.g., feature selection, semi-supervised classification, semi-supervised clustering, partially supervised classification, and model selection.

Chapter 3 presents an unsupervised sense discrimination method that can automatically determine an optimal feature subset and sense number for a target word. Moreover, it is empirically compared with another state of the art method for sense discrimination on benchmark corpora.

Chapter 4 provides an investigation of a graph based semi-supervised learning algorithm for sense disambiguation. Moreover, we empirically compare it with other related sense disambiguation methods, e.g., SVM, bootstrapping, and co-training.

Chapter 5 describes a partially supervised sense disambiguation method and empirically compare it with other related algorithms on benchmark corpora, e.g., a one-class classification algorithm (LPU), and a clustering based order identification method.

Some of the material presented in this thesis has been published. This applies to chapter 3 (ACL 2004), chapter 4 (ACL 2005), and chapter 5 (EMNLP 2006).

# Chapter 2

# Literature Review on Related Work

## 2.1 Feature Selection

Feature selection is to identify the most effective subset from the original features, while feature extraction transforms the original features to generate new salient features, e.g. feature clustering.

There is a long history of feature selection techniques for supervised learning in machine learning. Many approaches have been proposed to deal with the supervised feature selection problem. They can be categorized as filter approaches and wrapper approaches. Supervised filters conduct feature subset selection as a preprocessing step without considering the effects of selected feature subset on the performance of induction algorithm. Typically they measure the correlation of each feature with class label using distance, entropy, or dependence measures [31]. In wrapper methods for supervised learning, feature selection algorithms use induction algorithm as a black box to help evaluate each possible feature subset. Usually the prediction accuracy on class labels of the training data is a part of the evaluation function. Both filter and wrapper methods proposed for supervised learning use class labels to evaluate feature subsets.

But in unsupervised learning there is no class label on the dataset or the class label cannot be accessed by unsupervised learner, so the feature selection methods proposed for supervised learning are not applicable for unsupervised learning.

Feature selection is important to the performance of a clustering algorithm because irrelevant features hamper the clustering algorithm to find the intrinsic structure from datasets. So feature selection can improve the description or prediction ability of the clustering algorithm. Another merit of feature selection is to improve the efficiency of clustering process. The evaluation functions for supervised learning are not applicable to unsupervised learning since unsupervised learner cannot access class labels in datasets. Another difficulty of unsupervised feature selection is that the correct number of clusters is usually unknown in advance and the optimal feature subset and optimal cluster number are inter-related.

Recently several methods have been presented to deal with the feature selection problem in unsupervised learning. All feature selection algorithms that do not use class labels to evaluate feature subsets can be used for unsupervised learning.

Feature filter for unsupervised learning does not utilize a clustering algorithm to help

evaluate feature subsets. They usually evaluate feature subsets using measures dependant on the intrinsic property of a dataset. The following methods fall into this category:

Talavera (2000) presented a feature filter algorithm for clustering on symbolic data, which was based upon the assumption that features are likely to be irrelevant if they are little correlated with other features in a dataset.

Mitra et al. (2002) introduced a feature similarity measure that evaluates how closely two features are related by the eigenvalues of a covariance matrix. Their algorithm can determine a set of maximally independent features by discarding the redundant ones based on a pairwise feature similarity measure.

Dash et al. (2002) proposed an entropy measure to evaluate the importance of feature subsets. The filter method determined an optimal feature subset via minimizing the value of entropy measure on a dataset, which was independent of the subsequent clustering process. Their experiment results on synthetic and real datasets showed that their filter can correctly find the most important subsets.

In wrapper methods for unsupervised learning, feature selection algorithm searches a good feature subset by incorporating evaluation of clustering result as part of their objective function.

Devaney and Ram (1997) described an unsupervised feature wrapper for clustering on symbolic data, where each feature subset was wrapped around the COBWEB clustering algorithm. Category utility of the resulting concept hierarchy was used as an evaluation criterion of feature subsets. The feature subset which maximized the evaluation criterion was chosen as the optimal one.

Agrawal et al. (1998) proposed a CLIQUE algorithm that can identify dense clusters in subspaces of maximum dimensionality. Their algorithm is able to discover clusters in different lower dimensional subspaces. Their algorithm can help improve the description ability of the clustering algorithm.

Vaithyanathan and Dom (1999) presented a Bayesian approach to find the number of clusters and important feature subsets. They used stochastic complexity as the model selection criterion. Then they compared the Bayesian criterion with a cross-validation based criterion for document clustering. Their experiment result indicated that the Bayesian criterion can select a better feature subset based on a mutual information performance criterion.

Dash and Liu (2000) proposed to rank features according to their importance on clustering based on entropy measure. Then a subset of important features was selected by wrapping the sorted features on a k-means algorithm to maximize a cluster separability criterion.

Dy and Brodley (2000) introduced a wrapper framework for feature subset selection using expectation-maximization clustering with order identification. They compared two feature selection criteria on synthetic and real-world datasets: maximum likelihood and scatter separability, which were different from the objective function for order identification. Their experiment results indicated that maximum likelihood prefers feature subsets whose cluster structures fit a gaussian mixture model, while scatter separability prefers feature subsets which lead cluster centroids far apart.

Kim et al. (2000) investigated feature subset selection on a k-means algorithm using four criteria: cluster cohesiveness, cluster distance, penalty for increasing the cluster number and minimization of the selected feature subset. An evolutionary selection algorithm was suggested for searching in the feature space.

Table 2.1: The assumptions of various semi-supervised learning methods.

| Methods | Assumptions |
|---|---|
| mixture model,EM | generative mixture model |
| transductive SVM | low density region between classes |
| co-training | conditionally independent and redundant feature splits |
| graph based methods | labels smooth on graph |

Law et al. (2002) proposed to solve both feature selection and cluster number estimation simultaneously via the EM algorithm using a Minimum Message Length criterion. Their algorithm estimated both saliency of features and number of mixture components from unlabeled data without explicit search.

Yeung and Wang (2002) used a gradient decent technique to learn feature weights, which helps to reduce the uncertainty of the similarity matrix in similarity based clustering. Feature weighting can increase the separability of clusters and enhance the quality of similarity based decision making.

Modha and Spangler (2003) introduced a feature weighting algorithm for integrating multiple feature spaces in a k-means algorithm. Each data object was represented as a tuple of multiple feature vectors. Feature weighting was to assign a suitable distortion measure to each feature space. The optimal feature weighting was the one that yielded the clustering result with minimal intra-cluster dispersion and maximal inter-cluster dispersion.

## 2.2 Semi-Supervised Classification

This section focuses on semi-supervised classification, which is a special form of classification. Traditional classifiers use only labeled data (feature vector / label pairs) to learn models. Labeled examples are often labor-intensive and time consuming to obtain. Therefore, many semi-supervised learning algorithms have been proposed to address this problem by use of a large amount of unlabeled data that can be cheaply acquired, together with the labeled data, to build better classifiers, e.g. mixture model, transductive SVM, co-training, and graph based methods. Table 2.1 summarizes the assumptions underlying these semi-supervised algorithms [166]. Because semi-supervised learning requires less human effort and gives higher accuracy, it is of great interest both in theory and in practice.

### 2.2.1 Generative Model

Early work in semi-supervised learning assumes there are two classes, and each class has a Gaussian distribution. This amounts to assuming that the data is generated by a mixture model. With a large amount of unlabeled data, the mixture components can be identified with the expectation-maximization (EM) algorithm. One needs only a single labeled example per component to fully determine the label of each mixture. This model has been successfully applied to text categorization. Nigam et al. (2000) applied the EM algorithm [35] on mixture of multinomial for the task of text classification. They showed that the resulting classifiers perform better than those trained only from labeled data.

If the mixture model assumption is correct, unlabeled data is guaranteed to improve accuracy [22, 23, 115]. However if the assumption is not satisfied, unlabeled data may actually hurt accuracy. This has been observed by multiple researchers. Cozman et al. (2003) gave a formal derivation on how this might happen. Even if the mixture model assumption is correct, in practice EM is prone to local maxima. If a local maximum is far from the global maximum, unlabeled data may again hurt learning. Remedies include smart choice of starting point by active learning [102].

### 2.2.2 Self-Training

Self-training (or bootstrapping) is a commonly used technique for semi-supervised learning. It usually works as follows:

(1) train a classifier with initial labeled data;

(2) the classifier is then used to classify unlabeled data;

(3) typically the most confident unlabeled points, together with their predicted labels, are added to the labeled data;

(4) the classifier is re-trained with the augmented labeled data, and steps (2) to (4) are repeated.

This algorithm will stop if there is no unlabeled data available.

Note the classifier uses its own predictions to teach itself. One can imagine that a classification mistake can reinforce itself. Some algorithms try to avoid this by "unlearn" unlabeled points if the prediction confidence drops below a threshold. Self-training has been applied to several natural language processing tasks [51, 86, 87, 107, 118, 155].

### 2.2.3 Co-Training

As a semi-supervised learning method, the co-training algorithm [13, 92] is applicable to classification tasks if there are at least two distinct and independent views, and either view of the examples would be sufficient for learning if there are enough labeled data. Specifically, two learning algorithms are trained separately on each view, and each algorithm will perform classification on new unlabeled examples randomly selected from a large dataset, then the most confidently classified examples are added into the training set of the other algorithm, while maintaining the class distribution in the labeled data. This process is terminated after running multiple times or until unlabeled data is not available.

Nigam and Ghani (2000) performed extensive empirical experiments to compare co-training with generative mixture models and EM. Their results showed that co-training performs well if the conditional independence assumption indeed holds. In addition, it is better to probabilistically label the entire unlabeled data, instead of a few most confident data points. They named this paradigm co-EM. Finally, if there was no natural feature split, the authors created an artificial split by randomly breaking the feature set into two subsets. They showed that co-training with artificial feature split still helps, though not as much as before.

Co-training makes strong assumption on the splitting of features. Some works have been done to relax this assumption. Goldman and Zhou (2000) used two learners of different type but both takes the whole feature set, and essentially used one learner's high confidence data

points, identified with a set of statistical tests, in unlabeled data to teach the other learning and vice versa. Balcan et al. (2005) relaxed the conditional independence assumption with a much weaker expansion condition, and justified the iterative co-training procedure. Zhou and Li (2005) proposed tri-training which uses three learners. If two of them agree on the classification of an unlabeled point, the classification is used to teach the third classifier. This approach thus avoids the need of explicitly measuring label confidence of any learner. It can be applied to datasets without different views, or different types of classifiers. More generally, we can define learning paradigms that utilize the agreement among different learners. Co-training can be viewed as a special case with two learners and a specific algorithm to enforce agreement. Leskes (2005) presented a generalization error bound for semi-supervised learning with multiple learners, an extension to co-training. The author showed that if multiple learning algorithms are forced to produce similar hypotheses (i.e. to agree) given the same training set, and such hypotheses still have a low training error, then the generalization error bound is tighter. The unlabeled data was used to assess the agreement among hypotheses. The author proposed a new Agreement-Boost algorithm to implement the procedure.

### 2.2.4 Transductive SVM

A standard SVM uses only labeled data, and its goal is to find a maximum margin linear boundary in the Reproducing Kernel Hilbert Space. As an extension of the standard SVM, TSVM uses both labeled data and unlabeled data, and its goal is to find a labeling of the unlabeled data, so that a linear boundary has the maximum margin on both the original labeled data and the newly labeled data. The decision boundary has the smallest generalization error bound on the unlabeled data [142]. Intuitively, the unlabeled data guides the linear boundary away from dense regions. However finding the exact transductive SVM solution is NP-hard. Several approximation algorithms have been proposed and show positive results (see [57, 9]).

### 2.2.5 Graph-Based Methods

Graph-based semi-supervised methods define a graph where the nodes represent labeled and unlabeled examples in a dataset, and edges (may be weighted) reflect the similarity of examples. These methods usually assume label smoothness over the graph. Graph methods are nonparametric, discriminative, and transductive in nature.

Many graph-based methods can be viewed as estimating a function f on the graph. One wants f to satisfy two constraints at the same time: 1) it should be close to the given labels on the labeled nodes, and 2) it should be smooth on the whole graph. This can be expressed in a regularization framework where the first term is a loss function, and the second term is a regularizer. Several graph-based methods listed here are similar to each other. They differ in the particular choice of the loss function and the regularizer.

Blum and Chawla (2001) dealt with semi-supervised learning as a graph mincut (also known as st-cut) problem. In the binary case, positive labels act as sources and negative labels act as sinks. The objective is to find a minimum set of edges whose removal blocks all the flow from the sources to the sinks. The nodes connecting to the sources are then labeled positive, and those to the sinks are labeled negative. Equivalently mincut is the mode of

a Markov random field with binary labels (Boltzmann machine). The loss function can be viewed as a quadratic loss with infinity weight: $\infty \sum_{i \in L}(y_i - y_{i|L})^2$, so that the values on the labeled data are in fact fixed at their given labels. The regularizer is $1/2 \sum_{i,j} w_{i,j}(y_i - y_j)^2$. $w_{ij} = exp(-\frac{d_{ij}^2}{\sigma^2})$ 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 $\sigma$ is used to control the weight $W_{ij}$. The equality holds because the $y$'s take binary (0 and 1) labels. Putting the two together, mincut can be viewed as minimizing the function

$$\infty \sum_{i \in L}(y_i - y_{i|L})^2 + 1/2 \sum_{i,j} w_{i,j}(y_i - y_j)^2, \tag{2.1}$$

subject to the constraint $y_i \in 0, 1, \forall i$.

One problem with mincut is that it only gives a hard classification without confidence.

Blum et al. (2004) perturbed the graph by adding random noise to the edge weights. Mincut was applied to multiple perturbed graphs, and the labels were determined by a majority vote. The procedure is similar to bagging, and creates a "soft" mincut. They empirically compared plain mincut [14], randomized mincut, Gaussian fields [165], and spectral graph transducer [59] on 20 newsgroup and UCI data. Experiment results showed that on 20 newsgroup data, randomized mincut and Gaussian fields perform comparably, and both of them outperform the other two methods, while on UCI data, plain mincut and Gaussian fields perform comparably, and both of them outperform the other two methods.

The Gaussian random fields and harmonic function method in [165] is a continuous relaxation to the difficulty of discrete Markov random fields (or Boltzmann machines). It can be viewed as having a quadratic loss function with an infinity weight, so that the labeled data are clamped (fixed at given label values), and a regularizer based on the graph combinatorial Laplacian $\triangle$:

$$E(f) = \infty \sum_{i \in L}(f_i - y_{i|L})^2 + 1/2 \sum_{i,j} w_{i,j}(f_i - f_j)^2 = \infty \sum_{i \in L}(f_i - y_{i|L})^2 + f^T \triangle f. \tag{2.2}$$

Notice $f_i \in R$, which is the key relaxation to Mincut. The minimum energy function $f = argmin_{f_L = Y_L} E(f)$ is harmonic; namely, it satisfies $\Delta f = 0$ on the unlabeled data points $U$, and is equal to $Y_L$ on the labeled data points $L$. The harmonic property means that the value of $f(i)$ at each unlabeled data point i is the average of its neighbors j's in the graph:

$$f_i = \frac{1}{D_{ii}} \sum_{j \sim i} w_{ij} f_j, for i \in U. \tag{2.3}$$

The solution is given by

$$f_U = (I - T_{UU})^{-1} T_{UL} Y_L \tag{2.4}$$

where $T = D^{-1}W$, $D_{ii} = \sum_j w_{ij}$, and $D_{ij} = 0$ if $i \neq j$.

Label propagation algorithm [164] may be considered as a variant of this Gaussian random fields and harmonic function method. The only difference is that the label propagation algorithm uses $\overline{T}$ to replace $T$ in its final solution, where $\overline{T}_{ij} = T_{ij} / \sum_{k=1}^{n} T_{ik}$.

Local and global consistency method [161] uses the loss function $\sum_{i=1}^{n}(f_i - y_{i|L})^2$, and the normalized Laplacian $D^{-1/2}\Delta D^{-1/2} = I - D^{-1/2}WD^{-1/2}$ in the regularizer,

$$1/2\sum_{ij} w_{ij}(f_i/\sqrt{D_{ii}} - f_j/\sqrt{D_{jj}})^2 = f_T D^{-1/2}\Delta D^{-1/2} f \tag{2.5}$$

The spectral graph transducer [59] can be viewed with a loss function and regularizer

$$min\ c(f-\gamma)^T C(f-\gamma) + f^T L f \tag{2.6}$$

subject to $f^T 1 = 0$ and $f^T f = n$, where $\gamma_i = \sqrt{l_-/l_+}$ for positive labeled data, $-\sqrt{l_+/l_-}$ for negative data, $l_-$ being the number of negative data and so on. $L$ can be the combinatorial or normalized graph Laplacian, with a transformed spectrum. $c$ is a weighting factor, and $C$ is a diagonal matrix for misclassification costs.

For more other semi-supervised models, see the survey of semi-supervised learning in [166].

## 2.3 Semi-Supervised Clustering

Semi-supervised clustering (or clustering with side information) performs clustering with prior knowledge as must-links (two points must be in the same cluster) and cannot-links (two points cannot be in the same cluster) [146]. The prior knowledge provides a limited form of supervision, too far from being representative of a target classification of the items, so that supervised learning is not possible, even in a transductive form. Note that class labels can always be translated into pairwise constraints for the labeled data items and, reciprocally, by using consistent pairwise constraints for some items one can obtain groups of items that should belong to the same cluster.

Semi-supervised clustering is a tension between satisfying these constraints and optimizing the original clustering criterion (e.g. minimizing the sum of squared distances within clusters). Procedurally one can adapt distance metric or cost function [11, 64, 153] to try to accommodate the constraints, or one can bias the search [6, 146].

## 2.4 Learning with Positive and Unlabeled Examples

In many real world applications, labeled data may be available from only one of the two classes, and there is a large amount of unlabeled data that contains data for both classes. There are two ways to formulate this problem: classification or ranking.

### 2.4.1 Classification

Here one builds a classifier even though there is no negative example. It is important to note that with the positive training data one can estimate the positive class conditional probability $p(x|+)$, and with the unlabeled data one can estimate $p(x)$. If the prior $p(+)$ is known or estimated from other sources, one can derive the negative class conditional

probability as $p(x|-) = (p(x) - p(+)p(x|+))/(1 - p(+))$. With $p(x|-)$ one can then perform classification with Bayes rule. Denis et al. (2002) used this fact for text classification with Naive Bayes models.

Lee and Liu (2003) transformed the problem of learning with positive and unlabeled examples into a problem of learning with noise by labeling all unlabeled examples as negative and using a linear function to learn from the noisy examples. To learn a linear function with noise, they performed logistic regression after weighting the examples to handle noise rates of greater than a half. With appropriate regularization, the cost function of the logistic regression problem is convex, allowing the problem to be solved efficiently. To select regularization parameters for logistic regression, they proposed a performance criterion that can be estimated from a validation set (held-out positive data+unlabeled data). Their experiments on a text classification corpus showed that the methods proposed are effective, compared with S-EM [79] and one-class SVM [124].

Another set of methods heuristically identify a set of reliable negative documents from the unlabeled data, and then build a classifier using learned positive and negative data [79, 80, 81, 160].

Manevitz and Yousef (2001) proposed one-class SVM based on identifying Outlier's data as representative of the second-class and compared it with one-class SVM by Scholkopf et al. (1999) that tries to learn the support of the positive distribution by the use of only positive data, one-class versions of the algorithms prototype (Rocchio), nearest neighbor, naive Bayes, and a natural one-class neural network classification method based on bottleneck compression generated filters. The SVM approach as represented by Scholkopf was superior to all the methods except the neural network one, where it was, although occasionally worse, essentially comparable. Moreover, the SVM methods seemed to be quite sensitive to the choice of representation and kernel.

Yu et al. (2002) presented an Mapping-Convergence (MC) algorithm which works as follows:

(1) build a positive feature set PF which contains words that occur in the positive set P more frequently than in the unlabeled set U;

(2) a document in U that does not have any positive feature in PF will be added into negative document set RN;

(3) train a SVM using P, RN, and classify U-RN;

(4) extract negative data from U-RN and put them into RN;

(5) iteratively run step (3) and (4) till U-RN is empty.

Their experiments showed that MC algorithm (with positive and unlabeled data) achieves classification accuracy as high as that of traditional SVM (with positive and negative data) when the M-C algorithm uses the same amount of positive examples as that of traditional SVM.

Liu et al. (2003) proposed a biased SVM algorithm and empirically compared it with PEBL [160], S-EM [79], Roc-SVM [75] and all the possible combinations of methods of two steps in previous literature, e.g. Spy, 1-DNF, Rocchio, and NB for step 1, EM, SVM, SVM with iteration (SVM-I), and SVM with iteration and classifier selection (SVM-IS) for step 2. Roc-SVM and [(Spy or Rocchio or NB in step 1) + (SVM or SVM-I or SVM-IS for step 2)] can achieve state of the art performance on Reuters and 20 Newsgroup data. Furthermore, the biased SVM performed better than previous methods on Newsgroup data

with the expense of efficiency due to running SVM a large number of times.

### 2.4.2 Ranking

Given a large collection of items, and a few query items, ranking orders the items according to their similarity to the queries. It is worth pointing out that graph-based semi-supervised learning can be modified for such settings.

Joachims (2002) formulated the problem of learning a ranking function over a finite domain in terms of empirical risk minimization. Furthermore, he presented a ranking Support Vector Machine algorithm that leads to a convex program and that can be extended to non-linear ranking functions.

Zhou et al. (2004) treated it as semi-supervised learning with positive data on a graph, where the graph induces a similarity measure, and the queries are positive examples. Data points are ranked according to their graph similarity to the positive training set.

Information retrieval is another standard technique under this setting, but we will not attempt to include it here.

## 2.5 Model Selection

Model selection is linked to model assessment, which is the problem of comparing different models, or model parameters, for a specific learning task. For example, feature selection, classifier selection, and parameter learning can be considered as the cases of model selection.

In model selection, the goal is to select the one, among a set of candidate models, that represents the closest approximation to the underlying process based on some measure. Choosing the model that best fits a particular set of observed data will not accomplish the goal. For instance, it is well known that a complex model with many parameters and highly nonlinear form can often fit data better than a simple model with few parameters even if the latter generated the data. This is called overfitting.

Avoiding overfitting is what every model selection method is set to accomplish. The idea behind model selection methods is to select a model that captures only the underlying phenomenon in data, not the noise in data. Since noise is idiosyncratic to a particular data set, a model that captures noise will make poor predictions about future events. This leads to the present-day gold standard of model selection, generalizability. Generalizability, or predictive accuracy, refers to a model's ability to predict the statistics of future, as yet unseen, data samples from the same process that generated data sample.

### 2.5.1 Supervised Learning

For supervised learning, the standard practical technique for model selection is cross validation.

K-fold cross validation is a commonly used cross validation method, which holds out a data subset each time:

1. Randomly split training set $X$ into $k$ disjoint subsets of $n/k$ training examples each ($n = |X|$). Let's call these subsets $X_1, ..., X_k$.

2. For each model $M_i$, we evaluate it as follows: For $j = 1, ..., k$ Train the model $M_i$ on $X_1 \cup ... \cup X_{j-1} \cup X_{j+1} \cup ...X_k$( train on all the data except $X_j$) to get some hypothesis $h_{ij}$. Test the hypothesis $h_{ij}$ on $X_j$ , to get $e_{X_j}(h_{ij})$. The estimated generalization error of model $M_i$ is then calculated as the average of the $e_{X_j}(h_{ij})$'s (averaged over j).

3. Pick the model $M_i$ with the lowest estimated generalization error, and retrain that model on the entire training set $X$. The resulting hypothesis is then output as our final answer.

A typical choice for the number of folds to use here would be $k = 10$. While the fraction of data held out each time is now $1/k$ - much smaller than before - this procedure may also be more computationally expensive than hold-out cross validation, since we now need train to each model $k$ times.

While $k = 10$ is a commonly used choice, in problems in which data is really scarce, sometimes we will use the extreme choice of $k = n$ in order to leave out as little data as possible each time. In this setting, we would repeatedly train on all but one of the training examples in $X$, and test on that held-out example. The resulting $n = k$ errors are then averaged together to obtain our estimate of the generalization error of a model. This method is called leave-one-out cross validation.

### 2.5.2 Semi-Supervised Learning

Cross validation is not applicable for semi-supervised learning since in the typical setting of semi-supervised learning, there are only very few labeled examples as training data.

In the context of graph based methods, Zhu (2005) presented three methods for weight matrix learning from labeled and unlabeled data, including evidence maximization, entropy minimization, and minimum spanning tree.

The author assumed the edge weights are parameterized with hyperparameter $\Theta$. For example the edge weights can be

$$w_{ij} = exp(-\sum_{d=1}^{D} \frac{(x_{i,d} - x_{j,d})^2}{\alpha_d^2}) \tag{2.7}$$

and $\Theta = \{\alpha_1, ..., \alpha_D\}$. To learn the weight hyperparameters in a Gaussian process, one can choose the hyperparameters that maximize the log likelihood: $\hat{\Theta} = argmax_\Theta logp(y_L|\Theta)$. $logp(y_L|\Theta)$ is known as the evidence and the procedure is also called evidence maximization. One can also assume a prior on $\Theta$ and find the maximum a posteriori (MAP) estimate $\hat{\Theta} = argmax_\Theta(logp(y_L|\Theta) + logp(\Theta))$. The evidence can be multimodal and usually gradient methods are used to find a mode in hyperparameter space.

An alternative method for parameter learning is average label entropy. The average label entropy H(f) of the harmonic function f is defined as

$$H(f) = \frac{1}{u} \sum_{i=l+1}^{l+u} H_i(f(i)) \tag{2.8}$$

where $H_i(f(i)) = -f(i)logf(i) - (1 - f(i))log(1 - f(i))$ is the Shannon entropy of individual unlabeled data point $i$. Note that $0 \leq f(i) \leq 1$ for $i \in U$. Small entropy implies that

$f(i)$ is close to 0 or 1; this captures the intuition that a good $W$ (equivalently, a good set of hyperparameters $\Theta$) should result in a confident labeling.

For avoiding a complication, namely $H$ has a minimum at 0 as $\alpha_d \to 0$, the author smoothed the transition matrix $T$ with the uniform matrix $U$: $U_{ij} = 1/n$. The smoothed transition matrix is $\tilde{P} = \epsilon U + (1-\epsilon)P$. Then the author used gradient descent to find the hyperparameters $\alpha_d$ that minimize $H$.

The third method for weight matrix learning is to construct a minimum spanning tree over all data points with Kruskal's algorithm. In the beginning no node is connected. During tree growth, the edges are examined one by one from short to long. An edge is added to the tree if it connects two separate components. The process repeats until the whole graph is connected. The author found the first tree edge that connects two components with different labeled points in them. The author regarded the length of this edge $d^0$ as a heuristic to the minimum distance between different class regions. The author then set $\alpha = d^0/3$ following the $3\sigma$ rule of Normal distribution, so that the weight of this edge is close to 0, with the hope that local propagation is then mostly within classes.

### 2.5.3 Partially Supervised Learning

Lee and Liu (2003) proposed a performance criterion $pr/P(Y(x) = 1) = r^2/P(f(x) = 1)$ for regularization parameter estimation, in the setting of partially supervised classification (with only positive data and unlabeled data). $p$ stands for precision, $r$ for recall, $P(X)$ for the probability of $X$ is true, $Y$ for the true label of input $x$, $f$ for the hypothesis. $r$ and $P(f(x) = 1)$ can be estimated from validation set (positive data+ unlabeled data). This performance measure is proportional to the square of the geometric mean of precision and recall. It has roughly the same behavior as the $F$ score in the sense that it is large when both $p$ and $r$ are large and is small if either one is small. $F$ score requires both positive data and negative data for estimation of $p$ and $r$, but it cannot be used in the setting of partially supervised classification since negative data is not available here.

### 2.5.4 Unsupervised Learning

The intuitively simplest way to measure generalizability is to estimate it directly from the data, using cross-validation [134]. In cross-validation, data set is split into two samples, the training sample $X_{tr}$ and the test sample $X_{te}$. The best-fitting parameters are estimated by fitting the model to $X_{tr}$ which we denote $\theta(X_{tr})$. The generalizability estimate is obtained by measuring the fit of the model to the test sample at those original parameters,

$$CV = lnP(X_{te}|\theta(X_{tr})). \tag{2.9}$$

The main attraction of CV is its ease of implementation. All that is required is a model fitting procedure and a resampling scheme. One concern with CV is that there is a possibility that the test sample is not truly independent of the training sample: Since both are produced in the same experiment, systematic sources of error variation are likely to induce correlated noise across the two samples, artificially inflating the CV measure.

An alternative approach is to use theoretical measures of generalizability based on a single

sample. In most of these theoretical approaches, generalizability is measured by suitably combining goodness-of-fit with model complexity. The practical difference between them is the way in which complexity is measured.

**AIC**

The Akaike information criterion (AIC) [3] quantifies the relative goodness-of-fit and complexity of various previous derived statistical models, given a sample of data. It treats complexity as the number of parameters, $k$. $\hat{\theta}$ denotes the estimated parameters from the input observations $X$.

$$AIC = -lnP(X|\hat{\theta}) + k; \tag{2.10}$$

The method prescribes that the model minimizing AIC should be chosen. AIC seeks to find the model that lies closest to the true distribution, as measured by the Kullback-Leibler distance. As shown in the above criterion equation, this is achieved by trading the first, minus goodness-of-fit term of the right hand side for the second complexity term. As such, a complex model with many parameters, having a large value of the complexity term, will not be selected unless its fit justifies the extra complexity.

**BIC**

Another approach is given by the much older notion of Bayesian statistics. In the Bayesian approach, we assume that a priori uncertainty about the value of model parameters is represented by a prior distribution $\pi(\theta)$. Upon observing the data $X$, this prior is updated, yielding a posterior distribution $\pi(\theta|X) \propto P(X|\theta)\pi(\theta)$. In order to make inferences about the model (rather than its parameters), we integrate across the posterior distribution. Under the assumption that all models are a priori equally likely (because the Bayesian approach requires model priors as well as parameter priors), Bayesian model selection chooses the model $M$ with highest marginal likelihood defined as: $P(X|M) = \int P(X|\theta)\pi(\theta)d\theta$. The ratio of two marginal likelihoods is called a Bayes factor (BF), which is a widely used method of model selection in Bayesian inference. The two integrals in the Bayes factor are nontrivial to compute unless $P(X|\theta)$ and $\pi(\theta)$ form a conjugated family. Monte Carlo methods are usually required to compute BF, especially for highly parameterized models. A large sample approximation of BF yields the easily-computable Bayesian information criterion (BIC) [127]

$$BIC = -lnP(X|\hat{\theta}) + \frac{k}{2}lnn. \tag{2.11}$$

$n$ is the size of $X$. The model minimizing BIC should be chosen. It is important to recognize that the BIC is based on a number of restrictive assumptions. If these assumptions are met, then the difference between two BIC values approaches twice the logarithm of the Bayes factor as $n$ approaches infinity.

**MDL**

The Minimum Description Length principle is a strategy (criterion) for data compression and statistical estimation, proposed by Rissanen (1978). MDL states that, for both data compression and statistical estimation, the best probability model with respect to given data is the one that requires the shortest code length in bits for encoding the model itself and the data observed through it. A series of papers by Rissanen expanded on and refined this idea, yielding a number of different model selection criteria (one of which was identical to

the BIC). The most complete MDL criterion currently available is the stochastic complexity (SC [121]) of the data relative to the model,

$$SC = -lnP(X|\hat{\theta}) + ln \int_{\hat{\theta}(Y)\in\Theta} P(Y|\hat{\theta}(Y))dY : \tag{2.12}$$

$\Theta$ represents a multi-dimensional Euclidean space. Note that the second term of SC represents a measure of model complexity. Since the integral over the sample space is generally non-trivial to compute, it is common to use the Fisher-information approximation (FIA [120]): Under regularity conditions, the stochastic complexity asymptotically approaches

$$FIA = -lnP(X|\hat{\theta}) + \frac{k}{2}ln(\frac{n}{2\pi}) + ln \int_{\Theta} \sqrt{detI(\theta)}d\theta \tag{2.13}$$

where $I(\theta)$ is the expected Fisher information matrix of sample size one, consisting of the covariances between the partial derivatives of $L$ with respect to the parameters. Once again, the integral can still be intractable, but it is generally easier to calculate than the exact SC. As in AIC and BIC, the first term of FIA is the lack of fit term and the second and third terms together represent a complexity measure.

When using generalizability measures, it is important to recognize that AIC, BIC and FIA are all asymptotic criteria, and are only guaranteed to work as $n$ becomes arbitrarily large, and when certain regularity conditions are met [96]. The AIC and BIC in particular can be misleading for small $n$. The FIA is safer (i.e., the error level generally falls faster as $n$ increases), but it can still be misleading in some cases. The SC and BF criteria are more sensitive, since they are exact rather than asymptotic criteria, and can be quite powerful even when presented with very similar models or small samples.

Cluster number estimation is an important model selection problem in unsupervised learning. Several procedures have been proposed for inferring the number of clusters in an unsupervised manner, making use of nothing more than the available unlabeled data.

**Gap Statistic**

Tibshirani et al. (2001a) proposed the Gap Statistic that is applicable to Euclidian data only. For a given number of clusters $k$, a dataset $X$ and clustering solution $Y = A_k(X)$, the total sum of within-cluster dissimilarities $W_k$ is computed.

$$W_k = \sum_{1\leq v\leq k} (\frac{1}{2n_v} \sum_{i,j:Y_i=Y_j=v} D_{ij}) \tag{2.14}$$

where $D_{ij}$ denotes the dissimilarity between $X_i$ and $X_j$ (squared Euclidean distances) and $n_v = |\{i|Y_i = v\}|$ the number of objects assigned to cluster $v$ by labeling $Y$. This quantity computed on the original data is compared with the average over data that are generated from reference distribution, which results in the Gap.

$$Gap_n(k) = E_n(log(W_k)) - log(W_k) \tag{2.15}$$

where $E_n$ is the expectation under a sample of size $n$ from reference distribution. The $k$ which maximizes the gap between these two quantities is the estimated number of clusters. This method assumes that the data is spherically distributed.

**Clest**

Recently, resampling-based approaches for model order selection have been proposed that perform model assessment in the spirit of cross validation. These approaches share the idea of prediction strength or replicability as a common trait. The methods exploit the idea that a clustering solution can be used to construct a predictor, in order to compute a solution for the second dataset and to compare the computed and predicted class memberships for the second dataset.

In an early study, Breckenridge (1989) investigated the usefulness of this approach (called replication analysis there) for the purpose of cluster validation. Although his work did not lead to a directly applicable procedure, in particular not for model order selection, his study suggested the usefulness of such an approach for the purpose of validation.

Fridlyand and Dudoit (2001) proposed a model order selection procedure, called Clest, that also builds upon Breckenridge's work. Their method employed the replication analysis idea by repeatedly splitting the available data into two parts. Free parameters of their method were the predictor, the measure of agreement between a computed and a predicted solution and a baseline distribution similar to the Gap Statistic. Because these three parameters largely influence the assessment, their proposal may be considered more as a conceptual framework than as a concrete model order estimation procedure.

**Prediction Strength**

Tibshirani et al. (2001b) formulated a Prediction Strength method for inferring the number of clusters which is based on using nearest centroid predictors. The main idea is to a) cluster the test data into k clusters; b) cluster the training data into k clusters, and then c) measure how well the training set cluster centers predict co-memberships in the test set. For each pair of test observations that are assigned to the same test cluster, they determine whether they are also assigned to the same cluster based on the training centers.

Randomly split data set $X$ into training data $X_{tr}$ and test data $X_{te}$. Denote the clustering operation on these two datasets by $C(X_{tr}, k)$ and $C(X_{te}, k)$, where $k$ is the possible value of cluster number. let $D[C(...), X_{tr}]$ be an $n_{tr} \times n_{tr}$ matrix, with $ij$-th element $D[C(...), X_{tr}]_{ij} =$ 1 if observations $i$ and $j$ fall into the same cluster, and zero otherwise. They call these entries co-memberships.

For a candidate number of clusters $k$ let $A_{k1}$, $A_{k2}$, ..., $A_{kk}$ be the indices of the test observations in test clusters 1, 2, ..., $k$. Let $n_{k1}$, $n_{k2}$, ..., $n_{kk}$ be the number of observations in these clusters. They define the "prediction strength" of the clustering $C(X_{tr}, k)$ by

$$ps(k) = min_{1 \le l \le k} \frac{1}{n_{kl}(n_{kl} - 1)} \sum_{i \neq j \in A_{kl}} D[C(X_{tr}, k), X_{te}]_{ij} \tag{2.16}$$

For each test cluster, they compute the proportion of observation pairs in that cluster that are also assigned to the same cluster by the training set centroids. The prediction strength is the minimum of this quantity over the $k$ test clusters.

If k is equal to the true number of clusters, then the $k$ training set clusters will be similar to the $k$ test set clusters, and hence will predict them well. They select the $k$ with PS score above a threshold as the answer.

**Levine and Domany (2001)'s Cluster Validation**

In the approach from Levine and Domany (2001), $r$ subsamples $X^{\mu}$ $(1 \le \mu \le r)$ of size $[fn]$ $(f \in [0, 1]$, $n = |X|)$ are drawn from the original data. The clustering is performed

on the entire dataset and on the r subsamples. A similarity criterion $\Phi$ is proposed for the comparison of clustering solutions between the full dataset and the subsamples. The $n$ by $n$ matrix $C$ with $C_{ij} = 1$ ($i \neq j$ and i, j are in the same cluster) and 0 otherwise where $i, j \in 1, ..., n$, is called the cluster connectivity matrix. The resampling results in $r$ such $fn \times fn$ matrices $C^{(1)}, ...C^{(r)}$. For the parameter $k$, the similarity criterion $\Phi$ is:

$$\Phi(k) = \frac{1}{r} \sum_{\mu=1}^{r} \frac{\sum_{i,j} 1\{C_{i,j}^{(\mu)} = C_{i,j} = 1, i, j \in X^{\mu}\}}{\sum_{i,j} 1\{C_{i,j} = 1, i, j \in X^{\mu}\}} \tag{2.17}$$

$\Phi(k)$ measures the proportion of data point pairs in each cluster computed on a full dataset that are also assigned into the same cluster by clustering solution on a data subset. Clearly, $0 \leq \Phi(k) \leq 1$. Intuitively, if cluster number $k$ is identical with the true value, then clustering results on different subsets generated by sampling should be similar with that on the full dataset. In other words, the clustering solution with true model order as parameter is robust against resampling, which gives rise to a local optimum of $\Phi$.

**Ben-Hur et al. (2002)'s Cluster Validation**

Given the data $X$ with size $n$, two subsamples are generated with size $fn$, where $f \in (0.5, 1)$. The solutions obtained for these subsamples are compared at the intersection of the sets. Their approach computes the similarity on the points common to both subsamples. The similarity measure used by the authors is the Fowlkes and Mallows measure of similarity. Let a labeling $L$ be a partition of $X$ into $k$ subsets $X^1 ... X^k$. If points $i$ and $j$ have the same labels, the connectivity matrix $C$ is 1 in the entry $ij$ ($C$ is a symmetric matrix of $fn \times fn$ entries), and 0 otherwise. To establish similarity between labelings, $L_1$ and $L_2$, of the two subsamples, a dot product is defined:

$$< L_1, L_2 > = < C^{(1)}, C^{(2)} > = \sum_{i,j} C_{ij}^{(1)} C_{ij}^{(2)} \tag{2.18}$$

This dot product computes the number of pairs of points clustered together. As the dot product, $< L_1, L_2 >$ satisfies the Cauchy-Schwartz inequality: $< L_1, L_2 > \leq \sqrt{< L_1, L_1 >< L_2, L_2 >}$, and thus can be normalized into a correlation or cosine similarity measure:

$$cor(L_1, L_2) = \frac{< L_1, L_2 >}{< L_1, L_1 >< L_2, L_2 >} \tag{2.19}$$

This is the Fowlkes and Mallows similarity measure.

**Stability**

Lange et al. (2002) proposed a stability criterion for supervised learning, which measured the disagreement between labels on training data and test data, both assigned by a predictor $g$:

$$S_{sup}(g) = E[\frac{1}{n} \sum_{i=1}^{n} 1\{g_{Z_{train}}(X_{test}) \neq g_{Z_{test}}(X_{test})\}] \tag{2.20}$$

where $Z_{train} = \{X_{train}, Y_{train}\} = \{X_{train,1}, Y_{train,1}, ..., X_{train,n_{train}}, Y_{train,n_{train}}\}$, and $Z_{test} = \{X_{test}, Y_{test}\} = \{X_{test,1}, Y_{test,1}, ..., X_{test,n_{test}}, Y_{test,n_{test}}\}$. $X$ are the objects and $Y$ are the labels.

This stability measures the self-consistency of the predictor $g$. Practical evaluation of

this stability criterion amounts to 2-fold cross-validation. However, unlike cross-validation, stability can also be defined in settings where no label information is available in test data.

Furthermore, they extended this criterion for semi-supervised and unsupervised learning.

In the setting of semi-supervised learning, there is no enough labeled data for cross validation. They propose to generate more labeled data by assigning labels on $X_{unlabeled}$ using a predictor trained on $Z_{train}$. Let $Z_{unlabeled} = \{X_{unlabeled}, Y_{unlabeled}\} = \{X_{unlabeled}, g_{Z_{train}}(X_{unlabeled})\}$. Then

$$S_{semi}(g) = E[\frac{1}{n}\sum_{i=1}^{n} 1\{g_{Z_{unlabeled}}(X_{test}) \neq g_{Z_{train}}(X_{test})\}] \tag{2.21}$$

For unsupervised learning, another problem arises. Since no specific label values are prescribed for the classes, label indices might be permuted from one instance to another, even when the partitioning is identical. For example, keeping the same label set, exchanging the class labels 1 and 2 leads to a new partitioning, which is not structurally different. In other words, label values are only known up to a permutation. In view of this non-uniqueness of the representation of a partitioning, they defined the permutation relating indices on the first set to the second set by the one which maximizes the agreement between the classes. The stability then reads

$$S_{un}(g) = E[min_{\pi\in\Omega_k}\frac{1}{n}\sum_{i=1}^{n} 1\{\pi(g_{Z_{train}}(X_{test})) \neq g_{Z_{test}}(X_{test})\}] \tag{2.22}$$

$Y_{train}$ and $Y_{test}$ are assigned by some clustering algorithm, which are used for training classifiers on $Z_{train}$ or $Z_{test}$. The authors also suggested the choices of classifiers in unsupervised learning. For example, k-means clustering suggests to use nearest centroid classification. Minimum spanning tree type clustering algorithms suggest nearest neighbor classifiers, and finally, clustering algorithms which fit a parametric density model should use the class posteriors computed by the Bayes rule for prediction.

The range of the stability $S_{un}(g)$ depends on $k$, therefore stability values cannot be compared for different values of $k$. The stability minimized over $\Omega_k$ is bounded from above by $1 - 1/k$, since for a larger instability, there exists a relabeling which has smaller stability costs. This stability value is asymptotically achieved by the random predictor $\rho_k$ which assigns uniformly drawn labels to objects. Normalizing $S$ by the stability of $\rho_k$ yields values independent of $k$. Thus the normalized stability criterion is defined as:

$$S^k_{un}(g) = S_{un}(g)/S_{un}(\rho_k) \tag{2.23}$$

In practice, the value of stability is estimated as average value of $S^k_{un}(g)$ from clustering results on multiple disjoint halves of full dataset.

Rabinovich (2005) provided an empirical comparison among six cluster validation criteria on three toy datasets. Figure 2.2 shows the results of estimated cluster numbers. We can see that Levine's method, Ben-Hur's method and Lange's method find the correct cluster numbers on two datasets, which outperform the other methods.

Table 2.2: Estimated cluster numbers on three datasets by various cluster validation criteria.

| Dataset | Levine | Gap Statistic | Prediction Strength | Ben-Hur | Clest | Stability | True k |
|---|---|---|---|---|---|---|---|
| 4 Gaussians | 2, 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| 3 spirals (k-means) | 6 | 1 | 1 | 6 | 10 | 6 | 3 |
| 3 spirals (path based) | 2, 3, 4 | 1 | 1 | 2, 3 | 1 | 3 | 3 |

# Chapter 3

# Word Sense Discrimination with Feature Selection and Order Identification Capabilities

Supervised sense disambiguation methods usually rely on the information from previous sense tagged corpus to determine the senses of words in an unseen text. They require a lot of sense tagged corpus, and heavily depend on manually compiled lexical resources. However, these lexical resources often miss domain specific word senses, and even many new words are not included inside. Learning word senses from untagged corpora may help us dispense with the need for an outside knowledge source for defining senses by only discriminating senses of words. A few algorithms [26, 108, 126] have been proposed to address the sense discrimination problem. But these sense discrimination algorithms require the cluster number to be provided. In practice, the value of cluster number or sense number is usually unknown in advance. The aim of this work is to present an algorithm to automatically estimate the sense number for sense discrimination.

This chapter is organized as follows. Section 3.1 introduces the word sense discrimination algorithm, which incorporates unsupervised feature selection and model order identification technique. Then section 3.2 provides the experimental results of this algorithm and discuss some findings from these results. Section 3.3 concludes this work and suggests some possible improvements.

## 3.1 Learning Procedure

Before providing the details of the learning algorithm, we will present the definition of word vectors, context vectors and sense vectors introduced in Schutze (1998).

### 3.1.1 Word Vectors

A vector for word $w$ is derived from the close neighbors of $w$ in a large corpus. Close neighbors are all the words that co-occur with $w$ in a sentence or a larger context. In the simplest case, the vector has an entry for each word that occurs in the corpus. The entry

for word $v$ in the vector for $w$ records the number of times that word $v$ occurs close to $w$ in the corpus. This is the vector space that Schutze refers to as Word Space, where word $v$ serves as a dimension in this space. Word vector in Word Space captures the typical topic or subject matter of a word.

### 3.1.2 Context Vectors

A context vector of a target word $t$ is the centroid (or sum) of the vectors of the words ($w$) occurring in the context of $t$'s occurrence. The centroid "averages" the direction of a set of word vectors. If many of the words in a context have a strong component for one of the topics, then the average of the vectors, the context vector, will also have a strong component for the topic. Conversely, if only one or two words represent a particular topic, then the context vector will be weak on this component. The context vector hence represents the strength of different topical or semantic components in a context. In the computation of the context vector, we may weight a word vector according to its discriminating potential using idf.

### 3.1.3 Sense Vectors

Sense representations are computed as groups of similar contexts. All the contexts of the target word are collected from the corpus. For each context, a context vector is computed. This set of context vectors is then clustered into a predetermined number of groups. Each group is considered as a sense vector.

### 3.1.4 Feature Selection

We divide the sense discrimination problem into two sub-problems, unsupervised feature selection and clustering analysis with order identification. Feature selection for word sense discrimination is to find important contextual words that help to discriminate senses of a target word without using class labels in a dataset. This problem can be generalized as selecting important feature subset in an unsupervised manner.

We propose a cluster validation based unsupervised feature subset evaluation method. Cluster validation has been used to solve the model order identification problem [65, 73]. Table 3.1 provides the feature subset evaluation algorithm. If some features in a feature subset are noisy, then the estimated cluster structure on data subset in selected feature space is not stable, which is more likely to be the artifact of random splitting. Then the consistency between cluster structures estimated from disjoint data subsets will be lower. Otherwise the estimated cluster structures should be more consistent. Here we assume that splitting does not eliminate some of the underlying modes in a dataset.

For the comparison of different clustering structures, predictors are constructed based on these clustering solutions, and then we use these predictors to classify the same data subset. The agreement between class memberships computed by different predictors on the same data subset can be used as the measure of consistency between cluster structures of different data subsets. We use the stability measure [65] (given in Table 3.1) to assess the agreement between class memberships.

For vector representation of each occurrence, one strategy is to construct its second order context vector by summing the vectors of contextual words, and then let the feature selection procedure start to work on these second order contextual vectors to select features. However, since the sense associated with a word's occurrence is always determined by very few feature words in its contexts, it is always the case that there exist more noisy words than the real features in the contexts. So, simply summing the contextual word's vectors together may result in noise-dominated second order context vectors.

To deal with this problem, we extend the feature selection procedure further to the construction of second order context vectors: to select better feature words in contexts to construct better second order context vectors enabling better feature selection.

Since the sense associated with a word's occurrence is always determined by some feature words in its contexts, it is reasonable to suppose that the selected features should cover most of occurrences. Formally, let $coverage(D, T)$ be the coverage rate of the feature set $T$ with respect to a set of contexts $D$, i.e., the ratio of the number of the occurrences with at least one feature in their local contexts against the total number of occurrences, then we assume that $coverage(D, T) \geq \tau$. In practice, we set $\tau = 0.9$.

This assumption also helps to avoid the bias toward the selection of fewer features, since with fewer features, there are more occurrences without features in contexts, and their context vectors will be zero valued, which tends to result in more stable cluster structure.

Let $X$ be a set of local contexts of occurrences of the target word, then $X = \{x_i\}_{i=1}^{N}$, where $x_i$ represents local context of the $i$-th occurrence, and $N$ is the total number of this word's occurrences.

$W$ is used to denote bag of words occurring in context set $X$, then $W = \{w_i\}_{i=1}^{M}$, where $w_i$ denotes a word occurring in $X$, and $M$ is the total number of different contextual words.

Let $V$ denote a $M \times M$ second-order co-occurrence symmetric matrix. Suppose that the $i$-th, $1 \leq i \leq M$, row in the second order matrix corresponds to word $w_i$ ($w_i \in W$) and the $j$-th, $1 \leq j \leq M$, column corresponds to word $w_j$ ($w_j \in W$), and then the entry specified by $i$-th row and $j$-th column records the number of times that word $w_i$ occurs close to $w_j$ in corpus. We use $v(w_i)$ to represent the word vector of contextual word $w_i$, which is the $i$-th row in the matrix $V$.

$H^T$ is a weight matrix of contextual word subset $T$, $T \subseteq W$. Then each entry $h_{i,j}$ represents the weight of word $w_j$ in $x_i$, $w_j \in T$, $1 \leq i \leq N$. We use binary term weighting method to derive context vectors: $h_{i,j} = 1$ if word $w_j$ occurs in $x_i$, otherwise zero.

Let $C^T = \{c_i^T\}_{i=1}^{N}$ be a set of context vectors in feature subset $T$, where $c_i^T$ is the context vector of the $i$-th occurrence. $c_i^T$ is defined as:

$$c_i^T = \sum_j (h_{i,j} v(w_j)), w_j \in T, 1 \leq i \leq N. \tag{3.1}$$

The feature subset selection in word set $W$ can be formulated as:

$$\hat{T} = \arg\max_T \{criterion(T, H, V, q)\}, T \subseteq W, \tag{3.2}$$

subject to $coverage(X, T) \geq \tau$, where $\hat{T}$ is the optimal feature subset, *criterion* is a cluster validation based evaluation function (the function in Table 3.1), $q$ is resampling frequency

Table 3.1: An unsupervised Feature Subset Evaluation Algorithm. Intuitively, for a given feature subset $T$, we iteratively split the dataset into disjoint halves, and compute the agreement of clustering solutions estimated from the two datasets using stability measure. The average of stability over $q$ resamplings is the estimation of the score of $T$.

| | |
|---|---|
| | Function criterion($T$, $H$, $V$, $q$) |
| | Input parameter: feature subset $T$, weight matrix $H$, second order co-occurrence matrix $V$, resampling frequency $q$; |
| (1) | $S_T = 0$; |
| | Construct $C^T$ using $T$, $H$, $V$; |
| (2) | For $i = 1$ to $q$ do |
| (2.1) | Randomly split $C^T$ into disjoint halves, denoted as $C_A^T$ and $C_B^T$; |
| (2.2) | Estimate GMM parameter and cluster number on $C_A^T$ using *Cluster*, and the parameter set is denoted as $\hat{\theta}_A$; The solution $\hat{\theta}_A$ can be used to construct a predictor $\rho_A$; |
| (2.3) | Estimate GMM parameter and cluster number on $C_B^T$ using *Cluster*, and the parameter set is denoted as $\hat{\theta}_B$, The solution $\hat{\theta}_B$ can be used to construct a predictor $\rho_B$; |
| (2.4) | Classify $C_B^T$ using $\rho_A$ and $\rho_B$; The class labels assigned by $\rho_A$ and $\rho_B$ are denoted as $L_A$ and $L_B$; |
| (2.5) | $S_T += max_\pi \frac{1}{\lvert C_B^T \rvert} \sum_i 1\{\pi(L_A(c_{Bi}^T)) = L_B(c_{Bi}^T)\}$, where $\pi$ denotes possible permutation relating indices between $L_A$ and $L_B$, and $c_{Bi}^T \in C_B^T$; |
| (3) | $S_T = \frac{1}{q} S_T$; |
| (4) | Return $S_T$; |

for estimation of stability, and $coverage(X, T)$ is proportion of contexts with occurrences of features from $T$. This constrained optimization results in a solution which maximizes the criterion and satisfies the given constraint at the same time. In this work we used sequential floating backward search (SFBS) [113] in sorted word list based on $\chi^2$ or local frequency criterion. This search algorithm starts with a full feature set and, for each step, the worst feature (concerning the criterion) is eliminated from the set, e.g., one step of sequential backward selection (SBS). This algorithm also verifies the possibility of improvement of the criterion if some feature is included. In this case, the best feature that satisfies some criterion function is included with the current feature set, e.g., one step of the sequential forward selection (SFS) is performed. Therefore, the SFFS proceeds dynamically decreasing and increasing the number of features until the desired is reached. We set the the number of SBS step and the number of SFS step in SFBS as one.

### 3.1.5 Clustering with Order Identification

After feature selection, we employ a Gaussian mixture modelling algorithm, *Cluster* [16], to estimate cluster structure and cluster number on the whole untagged data. Let $X = \{x_n\}_{n=1}^{N}$ be a set of $M$ dimensional vectors to be modelled by GMM. Assuming that this model has $K$ subclasses, let $\pi_k$ denote the prior probability of subclass $k$, $\mu_k$ denote the $M$ dimensional mean vector for subclass $k$, $R_k$ denote the $M \times M$ dimensional covariance matrix for subclass $k$, $1 \leq k \leq K$. The subclass label for pixel $x_n$ is represented by $y_n$. MDL criterion is used for GMM parameter estimation and order identification, which is given by:

$$MDL(K, \theta) = -\sum_{n=1}^{N} \log\left(p_{x_n|y_n}(x_n|\Theta)\right) + \frac{1}{2} L \log(NM), \tag{3.3}$$

$$p_{x_n|y_n}(x_n|\Theta) = \sum_{k=1}^{K} p_{x_n|y_n}(x_n|k, \theta)\pi_k, \tag{3.4}$$

$$L = K(1 + M + \frac{(M+1)M}{2}) - 1, \tag{3.5}$$

The log likelihood measures the goodness of fit of a model to data sample, while the second term penalizes complex model. This estimator works by attempting to find a model order with minimum code length to describe the data sample $X$ and parameter set $\Theta$.

If the cluster number is fixed, the estimation of GMM parameter can be solved using EM algorithm to address this type of incomplete data problem [35]. The initialization of mixture parameter $\theta^{(1)}$ is given by:

$$\pi_k^{(1)} = \frac{1}{K_o} \tag{3.6}$$

$$\mu_k^{(1)} = x_n, where\ n = \lfloor (k-1)(N-1)/(K_o - 1) \rfloor + 1 \tag{3.7}$$

$$R_k^{(1)} = \frac{1}{N} \Sigma_{n=1}^{N} x_n x_n^t \tag{3.8}$$

$K_o$ is a given initial subclass number, which is larger than the possible correct cluster number.

Then EM algorithm is used to estimate model parameters by minimizing MDL:

E-step: re-estimate the expectations based on previous iteration:

$$p_{y_n|x_n}(k|x_n, \theta^{(i)}) = \frac{p_{x_n|y_n}(x_n|k, \theta^{(i)})\pi_k}{\sum_{l=1}^{K}(p_{x_n|y_n}(x_n|l, \theta^{(i)})\pi_l)}, \tag{3.9}$$

M-step: estimate the model parameter $\theta^{(i)}$ to maximize the log-likelihood in MDL:

$$\overline{N}_k = \sum_{n=1}^{N} p_{y_n|x_n}(k|x_n, \theta^{(i)}) \tag{3.10}$$

$$\overline{\pi}_k = \frac{\overline{N}_k}{N} \tag{3.11}$$

$$\overline{\mu}_k = \frac{1}{\overline{N}_k} \sum_{n=1}^{N} x_n p_{y_n|x_n}(k|x_n, \theta^{(i)}) \tag{3.12}$$

$$\overline{R}_k = \frac{1}{\overline{N}_k} \sum_{n=1}^{N} (x_n - \overline{\mu}_k)(x_n - \overline{\mu}_k)^t p_{y_n|x_n}(k|x_n, \theta^{(i)}) \tag{3.13}$$

$$p_{x_n|y_n}(x_n|k, \theta^{(i)}) = \frac{1}{(2\pi)^{M/2}} |\overline{R}_k|^{-1/2} \exp\{\lambda\} \tag{3.14}$$

$$\lambda = -\frac{1}{2}(x_n - \overline{\mu}_k)^t \overline{R}_k^{-1}(x_n - \overline{\mu}_k) \tag{3.15}$$

The EM iteration is terminated when the change of $MDL(K, \theta)$ is less than $\epsilon$:

$$\epsilon = \frac{1}{100}(1 + M + \frac{(M+1)M}{2}) log(NM) \tag{3.16}$$

For inferring the cluster number, EM algorithm is applied for each value of $K$, $1 \leq K \leq K_o$, and the value $\hat{K}$ which minimizes the value of MDL is chosen as the correct cluster number. To make this process more efficient, two cluster pairs are selected to minimize the change in MDL criteria when reducing $K$ to $K - 1$. Then these two selected clusters are merged. The resulting parameter set is chosen as an initial condition for EM iteration with $K - 1$ subclasses. This operation will avoid a complete minimization with respect to $\pi$, $\mu$, and $R$ for each value of $K$.

## 3.2 Experiments and Evaluation

### 3.2.1 Test Data

We constructed four datasets from sense-tagged corpora, "hard", "interest", "line", and "serve"[1], by randomly selecting 500 instances for each ambiguous word. The details of these datasets are given in Table 3.2. The preprocessing included lowering the upper case characters, ignoring all words that contain digits or non alpha-numeric characters, removing words from a stop word list, and filtering out low frequency words which appeared only once in entire set. We did not use stemming procedure. The sense tags were removed when they were used by our algorithm, feature selection+GMM ($FSGMM$), and Schutze's context group discrimination method ($CGD$). In the evaluation procedure, the sense tags in these four datasets were used as ground truth classes. A second order co-occurrence matrix for English words was constructed using English version of Xinhua News (Jan. 1998-Dec. 1999). The window size for counting second order co-occurrence was 50 words.

### 3.2.2 Evaluation Method for Feature Selection

For evaluation of feature selection, we used mutual information between feature subset and class label set to assess the importance of selected feature subset. The assessment measure

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

Table 3.2: Four ambiguous words, their senses and frequency distribution of each sense.

| Word | Sense | Percentage |
|---|---|---|
| hard | not easy (difficult) | 82.8% |
| (adjective) | not soft (metaphoric) | 9.6% |
| | not soft (physical) | 7.6% |
| interest | money paid for the use of money | 52.4% |
| | a share in a company or business | 20.4% |
| | readiness to give attention | 14% |
| | advantage, advancement or favor | 9.4% |
| | activity that one gives attention to | 3.6% |
| | causing attention to be given to | 0.2% |
| line | product | 56% |
| (noun) | telephone connection | 10.6% |
| | written or spoken text | 9.8% |
| | cord | 8.6% |
| | division | 8.2% |
| | formation | 6.8% |
| serve | supply with food | 42.6% |
| (verb) | hold an office | 33.6% |
| | function as something | 16% |
| | provide a service | 7.8% |

is defined as:

$$M(T) = \frac{1}{|T|} \sum_{w \in T} \sum_{l \in L} p(w,l) log \frac{p(w,l)}{p(w)p(l)}, \tag{3.17}$$

where $T$ is the feature subset to be evaluated, $T \subseteq W$, $L$ is class label set, $p(w,l)$ is the joint distribution of two variables $w$ and $l$, $p(w)$ and $p(l)$ are marginal probabilities. $p(w,l)$ is estimated based on contingency table of contextual word set $W$ and class label set $L$. Intuitively, if $M(T_1) > M(T_2)$, $T_1$ is more important than $T_2$ since $T_1$ contains more information about $L$.

### 3.2.3 Evaluation Method for Clustering Result

Accuracy is a commonly used evaluation measure for supervised sense disambiguation. It measures the agreement between labeling results and hand-tagged sense labels in benchmark corpora. But for unsupervised sense disambiguation, we will encounter the difficulty that there is no sense tag for instances in each cluster. Therefore, we will employ the method used in [65] to assign different sense tags to only $min(|U|, |C|)$ clusters by maximizing the accuracy, where $|U|$ is the number of clusters, and $|C|$ is the number of ground truth classes. The underlying assumption here is that each cluster is considered as a class, and for any two clusters, they do not share the same class label. At most $|C|$ clusters are assigned sense tags, since there are only $|C|$ classes in benchmark data.

Given the contingency table $Q$ between clusters and ground truth classes, each entry $Q_{i,j}$

gives the number of occurrences which fall into both the $i$-th cluster and the $j$-th ground truth class. If $|U| < |C|$, we constructed empty clusters so that $|U| = |C|$. Let $\Omega$ represent a one-to-one mapping function from $C$ to $U$. It means that $\Omega(j_1) \neq \Omega(j_2)$ if $j_1 \neq j_2$ and vice versa, $1 \leq j_1, j_2 \leq |C|$. Then $\Omega(j)$ is the index of the cluster associated with the $j$-th class. Searching a mapping function to maximize the accuracy of $U$ can be formulated as:

$$\hat{\Omega} = \arg\max_{\Omega} \sum_{j=1}^{|C|} Q_{\Omega(j),j}. \tag{3.18}$$

Then the accuracy of the solution $U$ is given by

$$Accuracy(U) = \frac{\sum_j Q_{\hat{\Omega}(j),j}}{\sum_{i,j} Q_{i,j}}. \tag{3.19}$$

In fact, $\sum_{i,j} Q_{i,j}$ is equal to $N$, the number of occurrences of the target word in the dataset.

### 3.2.4 Experiments and Results

For each dataset, we tested following procedures:

$CGD_{term}$: We implemented the context group discrimination algorithm. Top $max(|W| \times 20\%,\ 100)$ words in contextual word list was selected as features using frequency or $\chi^2$ based ranking. Then k-means clustering[2] was performed on the context vector matrix using normalized Euclidean distance. K-means clustering was repeated 5 times and the partition with the best quality was chosen as final result. The number of clusters used by k-means clustering was set to be identical with the number of ground truth classes as done in the work by Schutze (1998). We ran $CGD_{term}$ with various word vector weighting methods when deriving context vectors, ex. $binary$, $idf$, $tf \cdot idf$.

$CGD_{SVD}$: The context vector matrix was derived using the same method in $CGD_{term}$. Then context vectors were reduced to 100 dimensions using SVD. If the dimension of context vector was less than 100, all of latent semantic vectors with non-zero eigenvalue were used for subsequent clustering. Then k-means clustering was conducted on the latent semantic space using normalized Euclidean distance. We also ran it with different weighting methods, e.g., $binary$, $idf$, $tf \cdot idf$. The number of clusters used by k-means clustering was equal to the number of ground truth classes.

$FSGMM$: Cluster validation based feature selection was conducted in the feature set used by $CGD$. Then the *Cluster* algorithm was used to group a target word's instances using Euclidean distance measure. $\tau$ was set as 0.90 in feature subset search procedure. The random splitting frequency is set as 10 for estimation of the score of feature subset. The initial subclass number was 20 and full covariance matrix was used for parameter estimation of each subclass.

For investigating the effect of different context window size on the performance of above three algorithms, we tested these three procedures with various context window sizes: $\pm 1$, $\pm 5$, $\pm 15$, $\pm 25$, and all of contextual words. The average length of sentences in 4 datasets is

[2]we used the k-means function in statistics toolbox of Matlab.

Table 3.3: Mutual information between selected feature subsets and class labels with $\chi^2$ based feature ranking.

| Word | Cont. wind. size | Sizes of feature subsets of CGD | MI of feature subsets of CGD $\times 10^{-2}$ | Sizes of feature subsets of FSGMM | MI of feature subsets of FSGMM $\times 10^{-2}$ | Comparison between MI values of CGD and FSGMM |
|---|---|---|---|---|---|---|
| hard | 1 | 18 | 6.4495 | 14 | 8.1070 | $<$ |
| | 5 | 100 | 0.4018 | 80 | 0.4300 | $<$ |
| | 15 | 100 | 0.1362 | 80 | 0.1416 | $<$ |
| | 25 | 133 | 0.0997 | 102 | 0.1003 | $<$ |
| | all | 145 | 0.0937 | 107 | 0.0890 | $>$ |
| interest | 1 | 64 | 1.9697 | 55 | 2.0639 | $<$ |
| | 5 | 100 | 0.3234 | 89 | 0.3355 | $<$ |
| | 15 | 157 | 0.1558 | 124 | 0.1531 | $<$ |
| | 25 | 190 | 0.1230 | 138 | 0.1267 | $<$ |
| | all | 200 | 0.1163 | 140 | 0.1191 | $<$ |
| line | 1 | 39 | 4.2089 | 32 | 4.6456 | $<$ |
| | 5 | 100 | 0.4628 | 84 | 0.4871 | $<$ |
| | 15 | 183 | 0.1488 | 128 | 0.1429 | $>$ |
| | 25 | 263 | 0.1016 | 163 | 0.0962 | $>$ |
| | all | 351 | 0.0730 | 192 | 0.0743 | $<$ |
| serve | 1 | 22 | 6.8169 | 20 | 6.7043 | $>$ |
| | 5 | 100 | 0.5057 | 85 | 0.5227 | $<$ |
| | 15 | 188 | 0.2078 | 164 | 0.2094 | $<$ |
| | 25 | 255 | 0.1503 | 225 | 0.1536 | $<$ |
| | all | 320 | 0.1149 | 244 | 0.1260 | $<$ |

32 words before preprocessing. The performance of above three algorithms on each dataset was assessed by equation 3.19.

Table 3.3 and 3.4 provide the scores of feature subsets selected by $FSGMM$ and $CGD$. Table 3.5 presents the average accuracy of three procedures with different feature ranking and weighting method. Each figure is the average over 5 different context window size and 4 datasets. We provide the detailed results of these three procedures in Figure 3.1.

From Table 3.3 and 3.4, we can see that $FSGMM$ achieves better score on mutual information (MI) measure than $CGD$ over 35 out of total 40 cases. This is the evidence that feature selection can remove noise and select important features.

As it was shown in Table 3.5, with both $\chi^2$ and $freq$ based feature ranking, $FSGMM$ algorithm performs better than $CGD_{term}$ and $CGD_{SVD}$ if we use average accuracy to evaluate their performance. Specifically, with $\chi^2$ based feature ranking, $FSGMM$ achieves 55.4% average accuracy, while the best average accuracy of $CGD_{term}$ and $CGD_{SVD}$ are 40.9% and 51.3% respectively. With $freq$ based feature ranking, $FSGMM$ achieves 51.2% average accuracy, while the best average accuracy of $CGD_{term}$ and $CGD_{SVD}$ is 45.1% and 50.2%.

Table 3.4: Mutual information between selected feature subsets and class labels with $freq$ based feature ranking.

| Word | Cont. wind. size | Sizes of feature subsets of CGD | MI of feature subsets of CGD $\times 10^{-2}$ | Sizes of feature subsets of FSGMM | MI of feature subsets of FSGMM $\times 10^{-2}$ | Comparison between MI values of CGD and FSGMM |
|---|---|---|---|---|---|---|
| hard | 1 | 18 | 6.4495 | 14 | 8.1070 | $<$ |
| | 5 | 100 | 0.4194 | 80 | 0.4832 | $<$ |
| | 15 | 100 | 0.1647 | 80 | 0.1774 | $<$ |
| | 25 | 133 | 0.1150 | 102 | 0.1259 | $<$ |
| | all | 145 | 0.1064 | 107 | 0.1269 | $<$ |
| interest | 1 | 64 | 1.9697 | 55 | 2.7051 | $<$ |
| | 5 | 100 | 0.6015 | 89 | 0.8309 | $<$ |
| | 15 | 157 | 0.2526 | 124 | 0.3495 | $<$ |
| | 25 | 190 | 0.1928 | 138 | 0.2982 | $<$ |
| | all | 200 | 0.1811 | 140 | 0.2699 | $<$ |
| line | 1 | 39 | 4.2089 | 32 | 4.4606 | $<$ |
| | 5 | 100 | 0.6895 | 84 | 0.7816 | $<$ |
| | 15 | 183 | 0.2301 | 128 | 0.2929 | $<$ |
| | 25 | 263 | 0.1498 | 163 | 0.2181 | $<$ |
| | all | 351 | 0.1059 | 192 | 0.1630 | $<$ |
| serve | 1 | 22 | 6.8169 | 20 | 7.0021 | $<$ |
| | 5 | 100 | 0.7045 | 85 | 0.8422 | $<$ |
| | 15 | 188 | 0.2763 | 164 | 0.3418 | $<$ |
| | 25 | 255 | 0.1901 | 225 | 0.2734 | $<$ |
| | all | 320 | 0.1490 | 244 | 0.2309 | $<$ |

Table 3.6 summarizes the automatically estimated cluster numbers by $FSGMM$ over 4 datasets. The estimated cluster number is $2 \sim 4$ for "hard", $3 \sim 6$ for "interest", $3 \sim 6$ for "line", and $2 \sim 4$ for "serve". It is noted that the estimated cluster numbers are less than the values of ground truth class number in most cases. It may be explained by following reasons: First, the data is not balanced, which may lead to that some important features cannot be retrieved. For example, the fourth sense of "serve", and the sixth sense of "line", their corresponding features are not up to the selection criteria. Second, some senses cannot be distinguished using only bag-of-words information, and their difference lies in syntactic information held by features. For example, the third sense and the sixth sense of "interest" may be distinguished by syntactic relation of feature words, while the bag of feature words occurring in their context are similar. Third, some senses are determined by global topics, rather than local contexts. For example, according to global topics, it may be easier to distinguish the first and the second sense of "interest". Moreover, we can see that the use of $frequency$ as feature selection ranking criterion results in getting the number of senses correct more often in comparison with $\chi^2$ based criterion.

Table 3.5: Average accuracy of three procedures with various settings over 4 datasets.

| Algorithm | Feature ranking method | Feature weighting method | Average accuracy |
|---|---|---|---|
| $FSGMM$ | $\chi^2$ | $binary$ | 0.554 |
| $CGD_{term}$ | $\chi^2$ | $binary$ | 0.404 |
| $CGD_{term}$ | $\chi^2$ | $idf$ | 0.407 |
| $CGD_{term}$ | $\chi^2$ | $tf \cdot idf$ | 0.409 |
| $CGD_{SVD}$ | $\chi^2$ | $binary$ | 0.513 |
| $CGD_{SVD}$ | $\chi^2$ | $idf$ | 0.512 |
| $CGD_{SVD}$ | $\chi^2$ | $tf \cdot idf$ | 0.508 |
| $FSGMM$ | $freq$ | $binary$ | 0.512 |
| $CGD_{term}$ | $freq$ | $binary$ | 0.451 |
| $CGD_{term}$ | $freq$ | $idf$ | 0.437 |
| $CGD_{term}$ | $freq$ | $tf \cdot idf$ | 0.447 |
| $CGD_{SVD}$ | $freq$ | $binary$ | 0.502 |
| $CGD_{SVD}$ | $freq$ | $idf$ | 0.498 |
| $CGD_{SVD}$ | $freq$ | $tf \cdot idf$ | 0.485 |

Figure 3.2 shows the average accuracy over three procedures in Figure 3.1 as a function of context window size for 4 datasets. For word "hard", the performance drops as window size increases, and the best accuracy(77.0%) was achieved at window size 1. For word "interest", sense discrimination does not benefit from large window size and the best accuracy(40.1%) is achieved at window size 5. For word "line", accuracy drops when window size increases, and the best accuracy(50.2%) is achieved at window size 1. For word "serve", sense discrimination performance benefits from large window size, and the best accuracy(46.8%) is achieved at window size 15.

Leacock et al. (1998) used Bayesian approach for sense disambiguation of three ambiguous words, "hard", "line", and "serve", based on cues from topical and local context. They observed that local context is more reliable than topical context as an indicator of senses for this verb and adjective, but slightly less reliable for this noun. Compared with their conclusion, our result is consistent with theirs for word "hard". But there are some differences for verb "serve" and noun "line". For word "serve", the possible reason is that we do not use position of local word and part of speech information, which may deteriorate the performance when local context($\leq 5$ words) is used. For word "line", the reason might come from selected feature subset that is not good when context window size is quite large.

## 3.3 Summary

Our word sense discrimination algorithm combines two novel ingredients: feature selection and model order identification. Feature selection is formalized as a constrained optimization problem, the output of which is a set of important features to determine word senses. Both cluster structure and cluster number are estimated by minimizing a MDL criterion.

Table 3.6: Automatically estimated mixture component number.

| Word | Context window size | Model order with $\chi^2$ | Model order with $freq$ |
|---|---|---|---|
| hard | 1 | 3 | 4 |
| | 5 | 2 | 2 |
| | 15 | 2 | 3 |
| | 25 | 2 | 3 |
| | all | 2 | 3 |
| interest | 1 | 5 | 4 |
| | 5 | 3 | 4 |
| | 15 | 4 | 6 |
| | 25 | 4 | 6 |
| | all | 3 | 4 |
| line | 1 | 5 | 6 |
| | 5 | 4 | 3 |
| | 15 | 5 | 4 |
| | 25 | 5 | 4 |
| | all | 3 | 4 |
| serve | 1 | 3 | 3 |
| | 5 | 3 | 4 |
| | 15 | 3 | 3 |
| | 25 | 3 | 3 |
| | all | 2 | 4 |

Compared with previous sense discrimination methods, this sense discrimination algorithm eliminates the requirement of specification of sense number. Furthermore, it incorporates a feature selection procedure which helps to improve the performance of sense discrimination.

Experimental results showed that this algorithm can retrieve important features, automatically estimate cluster number, and achieve better performance in terms of average accuracy than the $CGD$ algorithm which requires cluster number as input. This word sense discrimination algorithm is unsupervised in two folds: no requirement of sense tagged corpus, and no requirement of predefinition of sense number, which enables the automatic learning of word senses from raw texts.

The work in this chapter focuses on unsupervised sense disambiguation. However sense clustering results from unsupervised methods 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 untagged corpora and direct use of word senses, semi-supervised learning has received great attention recently. In next chapter, we will investigate a graph based semi-supervised learning method to use a large amount of untagged corpora, together with sense tagged corpus, to build a better sense tagger.

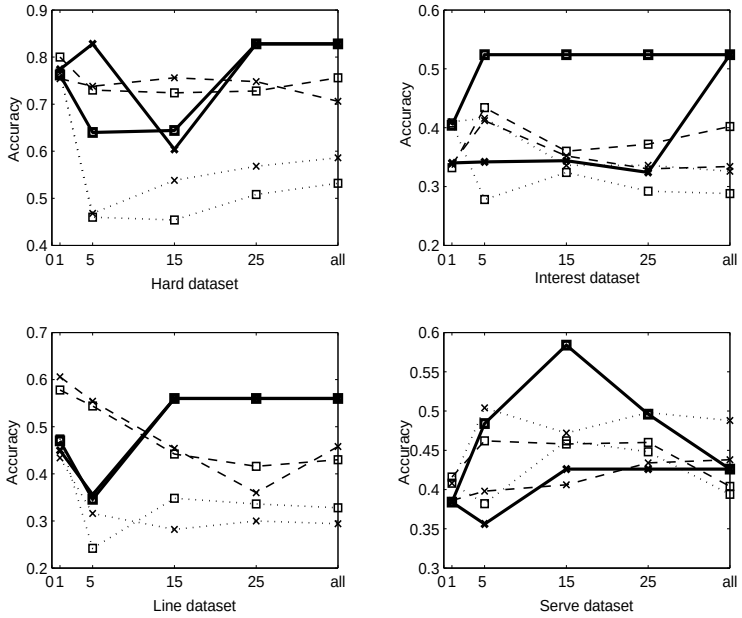


Figure 3.1: Results for three procedures over 4 datasets. The horizontal axis corresponds to the context window size. Solid line represents the result of $FSGMM + binary$, dashed line denotes the result of $CGD_{SVD} + idf$, and dotted line is the result of $CGD_{term} + idf$. Square marker denotes $\chi^2$ based feature ranking, while cross marker denotes $freq$ based feature ranking.

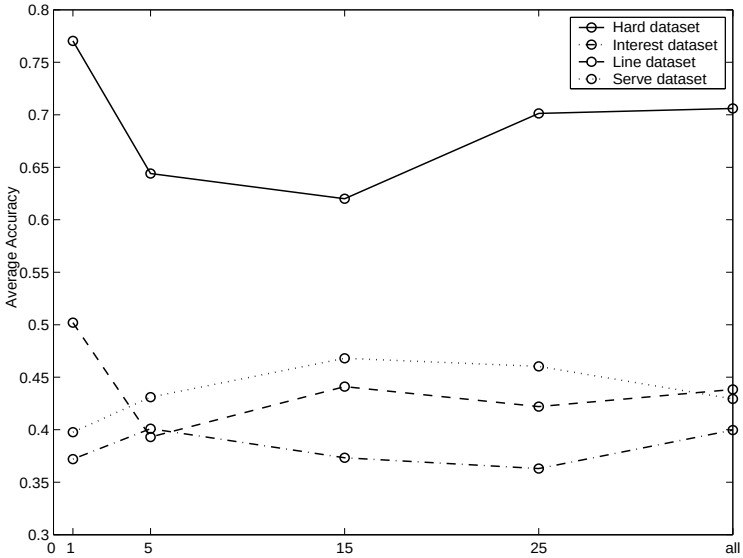


Figure 3.2: Average accuracy over three procedures in Figure 3.1 as a function of context window size (horizontal axis) for 4 datasets.

# Chapter 4

# Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning

Semi-supervised learning can use small labeled data and a large amount of unlabeled data to improve the performance of classifiers. As a promising family of techniques in semi-supervised learning, graph based methods represent all the data as a graph and try to estimate a labeling function to satisfy two constraints at the same time: 1) it should be close to the given labels on the labeled nodes, and 2) it should be smooth on the whole graph. This can be expressed in a regularization framework where the first term is a loss function, and the second term is a regularizer.

In this chapter we investigate a graph based semi-supervised learning method, the label propagation algorithm (LP), for sense disambiguation and empirically compare it with related supervised and semi-supervised sense disambiguation methods.

This chapter is organized as follows. First, we define the WSD problem in the context of semi-supervised learning in section 4.1. Then section 4.2 describes the LP algorithm and discuss the difference among SVM, bootstrapping and LP. Section 4.3 provides experimental results of LP algorithm and related algorithms on widely used benchmark corpora. Finally we conclude this work and suggest possible improvements in section 4.4.

## 4.1 Problem Setup

Let $X = \{x_i\}_{i=1}^{n}$ be a set of $n$ contexts 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 the 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 the context $x_h$ based on the labeling information of $x_g$ and the similarity information among examples in $X$.

The affinity among examples 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: $W_{ij} = exp(-\frac{d_{ij}^2}{\sigma^2})$ if $i \neq j$ and $W_{ii} = 0$ $(1 \leq i, j \leq n)$, where $d_{ij}$ is the distance (e.g., Euclidean distance) between $x_i$ and $x_j$, and $\sigma$ is used to control the weight $W_{ij}$.

## 4.2 Semi-Supervised Learning Method

### 4.2.1 A Label Propagation Algorithm

In the LP algorithm, the 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 the closer the examples are, the more likely they have similar labels.

We define soft label as a vector that is a probabilistic distribution over all the classes. In the 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. Hopefully, the values of $W_{ij}$ across different classes would be as small as possible and the values of $W_{ij}$ within the same class would be as large as possible. This will make label propagation to stay within the same class. This label propagation process will make the labeling function smooth on the graph.

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.

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 the example $x_j$ to the example $x_i$.

Compute the row-normalized matrix $\overline{T}$ by $\overline{T}_{ij} = T_{ij} / \sum_{k=1}^{n} T_{ik}$. This normalization is to maintain the class probability interpretation of $Y$. Then the LP algorithm is defined as follows:

Step 1 Initially set t=0, where $t$ is an iteration index;

Step 2 Propagate the label by $Y^{t+1} = \overline{T} Y^t$;

Step 3 Clamp labeled data by replacing the top $l$ row of $Y^{t+1}$ with $Y_L^0$. Repeat from step 2 till $Y^t$ converges;

Step 4 Assign $x_h (l + 1 \leq h \leq n)$ with label $s_{\hat{j}}$, where $\hat{j} = argmax_j Y_{hj}$.

### 4.2.2 Comparison between SVM, Bootstrapping and LP

For WSD, linear SVM is one of state of the art supervised learning algorithms [88], while bootstrapping is one of state of the art semi-supervised learning algorithms [74, 155]. To compare LP with SVM and bootstrapping, let us consider a dataset with two-moon pattern shown in Figure 4.1(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 other 20 points are unlabeled. The distance metric is Euclidian distance. We see that the points in one moon

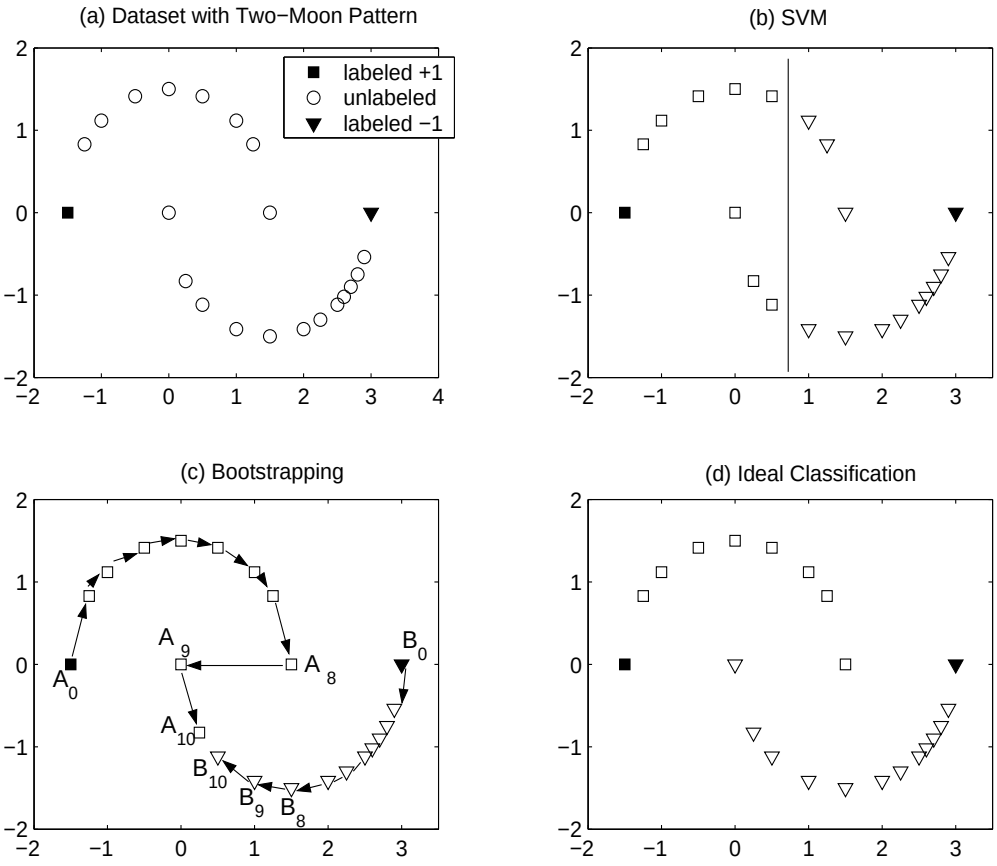


Figure 4.1: Classification result on a two-moon pattern dataset. (a) Two-moon pattern dataset with two labeled points, (b) classification result by SVM, (c) labeling procedure of bootstrapping algorithm, (d) ideal classification.

should be more similar to each other than the points across the moons from a point of global view as shown in Figure 4.1(d).

Figure 4.1(b) shows the classification result of the linear 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 see that SVM does not work well when the labeled data cannot reveal the structure (two moon pattern) in each class. The reason is that the classification hyperplane was learned only from the labeled data. In other words, the coherent structure (two-moon pattern) in the unlabeled data was not explored when inferring the class boundary.

Figure 4.1(c) shows a bootstrapping procedure using kNN (k=1) as a 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). Termination condition is that the distance between the labeled and unlabeled points is more than inter-class distance (the distance between $A_0$ and $B_0$). Each arrow in Figure 4.1(c) represents one classification operation in each iteration for each class. After eight iterations, $A_1 \sim A_8$ 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

labeled 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 the points in lower moon than to the points in upper moon. In other words, the bootstrapping method perform classification under a local consistency based strategy: the labels of unlabeled examples are determined only by labeled examples. This is the reason that two points $A_9$ and $A_{10}$ are misclassified by bootstrapping (shown in Figure 4.1(c)).

From the above analysis we can see that both SVM and bootstrapping are based on a local consistency assumption.

Finally, we ran LP on a connected graph, minimum spanning tree, generated for this dataset, shown in Figure 4.2(a). $A$, $B$, $C$ represent three points, and the edge $A - B$ connects the two moons. Figure 4.2(b)- 4.2(f) shows the convergence process of LP with $t$ increasing from 1 to 100. When $t = 1$, label information of labeled data was pushed to only nearby points. After seven iteration steps ($t = 7$), the point $B$ in the upper moon was misclassified as $-1$ since it first received label information from the point $A$ through the edge connecting the two moons. After another three iteration steps (t=10), this misclassified point was re-tagged as $+1$. The reason of this self-correcting behavior is that with the push of label information from nearby points, the value of $Y_{B,+1}$ became higher than that of $Y_{B,-1}$. In other words, the weight of the edge $B - C$ is larger than that of the edge $B - A$, which makes it easier for $+1$ label of the point $C$ to travel to the point $B$. Finally, when $t \geq 12$, LP converged to a fixed point, achieving the ideal classification result. In this label propagation process, we can see that LP uses the graph structure to smooth the labels of unlabeled examples.

## 4.3 Experiments and Results

### 4.3.1 Experiment Design

For empirical comparison with SVM, bootstrapping and co-training, we evaluated LP on widely used benchmark corpora - "interest", "line" [1], and the data in English lexical sample task of SENSEVAL-2 and SENSEVAL-3[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 [69] except that we did not use syntactic relations). For SVM, we did not perform feature selection on SENSEVAL-3 data since feature selection deteriorates its performance [69]. When running LP on the four datasets, we removed the features with occurrence frequency (counted in both training set and test set) less than 3 times.

We investigated two distance measures for LP: cosine similarity and Jensen-Shannon (JS) divergence [78]. Cosine similarity is a commonly used semantic distance, which measures the angle between two feature vectors. JS divergence has ever been used as distance measure for document clustering, which outperforms cosine similarity based document clustering [132]. JS divergence measures the distance between two probability distributions if feature vector is

[1]Available at http://www.d.umn.edu/∼tpederse/data.html

[2]Available at http://www.senseval.org/

Table 4.1: These two tables summarize accuracy (averaged over 20 trials) and paired t-test results of SVM and LP on SENSEVAL-3 corpus with the percentage of training set increasing from 10% to 100%.

| Percentage | SVM | $LP_{cosine}$ | $LP_{JS}$ |
|---|---|---|---|
| 10% | 53.4±1.1% | 55.0±1.3% | 56.2±1.2% |
| 25% | 62.3±0.7% | 64.2±0.7% | 65.5±0.7% |
| 50% | 66.6±0.5% | 66.7±0.5% | 68.2±0.4% |
| 75% | 68.7±0.4% | 67.5±0.3% | 69.5±0.3% |
| 100% | 69.7% | 68.0% | 69.8% |

| Percentage | SVM vs. $LP_{cosine}$ | | SVM vs. $LP_{JS}$ | |
|---|---|---|---|---|
| | p-value | Sign. | p-value | Sign. |
| 10% | 1.7e-8 | $\ll$ | 2.0e-13 | $\ll$ |
| 25% | 1.2e-11 | $\ll$ | 9.0e-18 | $\ll$ |
| 50% | 5.1e-1 | $\sim$ | 7.4e-13 | $\ll$ |
| 75% | 9.0e-10 | $\gg$ | 1.6e-8 | $\ll$ |
| 100% | - | - | - | - |

considered as probability distribution over features. Therefore we would like to select these two distance measures for sense disambiguation.

Let $JS(p, q)$ represent JS divergence between probability distribution $p(f)$ and $q(f)$ ($f$ is feature vector of an instance), which is defined as

$$JS(p, q) = \pi_p D_{KL}(p\|\overline{p}) + \pi_q D_{KL}(q\|\overline{p}), \tag{4.1}$$

$$D_{KL}(p\|\overline{p}) = \sum_f p log\frac{p}{\overline{p}}, \tag{4.2}$$

$$D_{KL}(q\|\overline{p}) = \sum_f q log\frac{q}{\overline{p}}, \tag{4.3}$$

$$\overline{p} = \pi_p p + \pi_q q, \tag{4.4}$$

where $\pi_p$ and $\pi_q$ are prior probabilities, and $\pi_p, \pi_q > 0, \pi_p + \pi_q = 1$. $D_{KL}$ is Kullback-Leibler distance, another measure of the distance between two probability distributions. However it is not a true metric since it is not symmetric and does not obey the triangle inequality.

For the four datasets, we constructed connected graphs as follows: two instances $x_i$ and $x_j$ will be connected by an edge if $x_i$ is among $x_j$'s k nearest neighbors, or if $x_j$ is among $x_i$'s k nearest neighbors as measured by cosine or JS distance measure. For all the datasets, k is set as 10 (following [165]). Moreover, we set $\sigma$ as the average distance between labeled examples from different classes.

### 4.3.2 Experiment 1: LP vs. SVM

In this experiment, we evaluated LP and SVM [3] on the data of English lexical sample task in SENSEVAL-3 (including all 57 English words ). We used $l$ examples from official training set as labeled data, and the remaining training examples and all the official 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 official training set. If any sense was absent from the sampled labeled set, we redid the sampling. We conducted experiments with different values of $l$, including $10\% \times N_{w,train}$, $25\% \times N_{w,train}$, $50\% \times N_{w,train}$, $75\% \times N_{w,train}$, and $100\% \times N_{w,train}$ ($N_{w,train}$ is the number of examples in training set of word $w$). SVM and LP were evaluated using accuracy [4] (fine-grained score) on official test set of SENSEVAL-3.

We conducted paired t-test on 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 $\leq 0.01$, $(0.01, 0.05]$, and $> 0.05$ respectively. $\ll$ and $<$ (or $\gg$ and $>$) means that the performance of SVM is significantly worse (or significantly better) than that of LP. $\sim$ means that the performance of SVM is almost as same as that of LP.

Table 4.1 reports the average accuracy and paired t-test results of SVM and LP with different sizes of labeled data [5].

From Table 4.1, we see that with very few labeled examples (the percentage of labeled data $\leq 25\%$), LP performs significantly better than SVM. When the percentage of labeled data increases from 50% to 75%, the performance of $LP_{JS}$ is still significantly better than that of SVM, while $LP_{cosine}$ performs worse than SVM. It seems that using the information of cluster structure in unlabeled data helps LP to locate the true class boundaries when labeled examples are not enough to reveal the structure in each class.

The performance of the top systems in ELS task of SENSEVAL-3 is around 71%. The LP method does not perform as good as the top systems in SENSEVAL-3, but there is some possible improvements to be done for this LP method, e.g., using more unlabeled examples.

### 4.3.3 Experiment 2: LP vs. Bootstrapping

Li and Li (2004) used "interest" and "line" corpora as test data for the evaluation of their algorithms. For word "interest", they used its four major senses. For comparison with their results, we took the same reduced "interest" corpus (constructed by retaining only four major senses) and whole "line" corpus as benchmark data. In their algorithm, $c$ is the number of senses of an ambiguous word, and $b$ ($b = 15$) is the number of examples added into classified data for each class in each iteration of bootstrapping. $c \times b$ may be considered as the size

[3] We used linear $SVM^{light}$ since linear SVM outperforms non-linear SVM for sense disambiguation [69]. $SVM^{light}$ is available at http://svmlight.joachims.org/.

[4] If there are multiple sense tags for an instance in training set or test set, then only the first tag is considered as correct answer. Furthermore, if the tag of the instance in test set is "U" (it is unassignable), then this instance will not be considered when calculating accuracy score.

[5] The accuracy reported here is slightly different from the results in [104] since here we set the value of k as '10', not '5' used in [104]

Table 4.2: Accuracy from [74] and average accuracy 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 | Accuracy from [74] | | | |
|---|---|---|---|---|
| | 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 | 4×15=60 | 80.2±2.0% | 79.8±2.0% |
| line | 6×15=90 | 60.3±4.5% | 59.4±3.9% |

of initial labeled data in their bootstrapping algorithm. We ran LP with 20 trials on the reduced "interest" corpus and whole "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 4.2 summarizes the average accuracy of LP on the two corpora. It also lists the accuracy of monolingual bootstrapping algorithm (MB) and bilingual bootstrapping algorithm (BB) on "interest" and "line" corpora. We 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.

### 4.3.4 Experiment 3: LP vs. Co-Training

As a semi-supervised learning method, co-training is applicable to classification tasks if there are at least two distinct and independent views, and either view of the examples would be sufficient for learning if there are enough labeled data. Specifically, two learning algorithms are trained separately on each view, and each algorithm will perform classification on new unlabeled $P$ examples randomly selected from a large dataset, then the most confidently classified $G$ examples are added into the training set of the other algorithm, while maintaining the class distribution in labeled data. This process is terminated after running $I$ times or till unlabeled data is not available.

Mihalcea (2004a) investigated an application of bootstrapping and co-training for WSD on the noun dataset in SENSEVAL-2. From the learning curves of bootstrapping and co-training with respect to the number of iterations, the author noticed that the curves usually consists of an increase of performance followed by a decline. Furthermore, there is no optimal value on the number of iterations across different words. This observation leads to a new method that combines co-training or bootstrapping with majority voting, which may improve the performance of learning methods by smoothing learning curves. In their experiments, examples with collocations that include the target word were removed from training data and test data. For comparison with Mihalcea's method, we did the same pre-processing

Table 4.3: Accuracy from [87] and the accuracy of LP on noun dataset in SENSEVAL-2. Major is a baseline method in which they always choose the most frequent sense. Basic bootstrapping denotes monolingual bootstrapping with Naive Bayes as base classifier. Basic co-training represents co-training using a local versus topical feature split, while Naive Bayes is used to implement local and topical classifiers. Smoothed co-training is an improvement of co-training algorithm by replacing the classifier in each iteration with a majority voting scheme applied to all classifiers constructed at previous iterations.

| | Accuracy from [87] | | | |
|---|---|---|---|---|
| Data set | Major | Basic bootstrapping | Basic co-training | Smoothed co-training |
| Nouns in SENSEVAL-2 | 53.8% | 54.2% | 55.7% | 58.4% |

| | Our results | |
|---|---|---|
| Data set | $LP_{cosine}$ | $LP_{JS}$ |
| Nouns in SENSEVAL-2 | 59.9% | 61.0% |

Table 4.4: Parameter values and accuracy of our re-implemented basic bootstrapping, smoothed bootstrapping, basic co-training, and smoothed co-training on the data of SENSEVAL-2 and SENSEVAL-3.

| | Our re-implementation | | | |
|---|---|---|---|---|
| | Basic bootstrapping | Smoothed bootstrapping | Basic co-training | Smoothed co-training |
| Value of G | 5 | 100 | 100 | 5 |
| Value of I | 5 | 5 | 5 | 50 |
| Accuracy on nouns in SENSEVAL-2 | 59.8% | 59.7% | 59.2% | 60.9% |
| Accuracy on all the data in SENSEVAL-3 | 67.6% | 68.2% | 63.6% | 65.0% |

on training and test data, and then ran LP algorithm on reduced datasets of nouns in SENSEVAL-2 data. The performance of LP algorithm was evaluated using accuracy on reduced test set.

Table 4.3 lists the results from Mihalcea (2004a) and the accuracy of LP algorithm on SENSEVAL-2 data. We see that LP outperforms basic bootstrapping, basic co-training, and smoothed co-training on this dataset.

### 4.3.5 Experiment 4: Re-Implementation of Bootstrapping and Co-Training

Mihalcea (2004a) applied bootstrapping and co-training on only nouns in SENSEVAL-2. In this work, we re-implemented the author's methods and evaluated them using all the data in English Sample task of SENSEVAL-3. The values of two parameters ($G$, $I$) in each learning approach were tuned by optimizing the performance on the noun dataset in SENSEVAL-2

respectively. The possible values of both $G$ and $I$ are 5, 10, 50, 100.

We used only test data as unlabeled data in this learning process. Since the size of test set is not very large, we did not select $P$ examples from unlabeled data to create a pool in each iterative process. Therefore the parameter $P$ was not used in our re-implementation. For bootstrapping and co-training, we used kNN as base classifier, and JS divergence as the distance measure. For co-training, we use local vs. topical features as feature split.

Table 4.4 lists the optimal values of $G$ and $I$ used by four methods (basic bootstrapping, smoothed bootstrapping, basic co-training, and smoothed co-training) and corresponding accuracy on the noun dataset in SENSEVAL-2. It shows that our re-implemented systems perform slightly better than Mihalcea's on the same data. Then we ran these systems on the data in English Sample task of SENSEVAL-3. Table 4.4 lists the accuracy of these four methods on the new data. The results in Table 4.1 and Table 4.4 indicate that using JS divergence as distance measure, LP algorithm achieved 69.8% accuracy on SENSEVAL-3 data, which outperforms basic bootstrapping, basic co-training and their variants using majority voting.

### 4.3.6 An Example: Word "use"

To investigate the reason for LP to outperform SVM, monolingual bootstrapping and co-training, we used the smallest dataset, the data of word "use", in English lexical sample task of SENSEVAL-3 as an example (totally 26 examples in training set and 14 examples in test set). For data visualization, we conducted unsupervised nonlinear dimensionality reduction[6] on these 40 feature vectors with 210 dimensions. Figure 4.3 (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 the training set and the test set served as unlabeled data for bootstrapping, co-training and LP. All of these three algorithms were evaluated using accuracy on test set.

From Figure 4.3 (c), we see that SVM misclassified many examples from class + to class $\times$ since using only features occurring in training data cannot reveal the intrinsic structure of the full data.

Moreover, we ran smoothed bootstrapping and smoothed co-training on this dataset to augment initial labeled data. Then kNN model was learned on the augmented labeled data and we used this model to perform classification on remaining unlabeled data. We used the same values of G and I for bootstrapping and co-training as in section 4.3.5. Figure 4.3 (d) and (e) report the final classification results of smoothed bootstrapping and smoothed co-training respectively, while Figure 4.3 (f) reports the classification result of LP algorithm.

In Figure 4.3 (d), A, B and C denote three labeled examples, while D, E, and F represent three examples that are correctly classified by LP, but misclassified by bootstrapping. Unlabeled example $D$ [7] happened to be closest to labeled example $A$, then kNN model tagged

[6]We used *Isomap* to perform dimensionality reduction by computing two-dimensional, 39-nearest-neighbor-preserving embedding of 210-dimensional input. *Isomap* is available at http://isomap.stanford.edu/.

[7]In the two-dimensional space, the example $D$ is not the closest example to $A$. The reason is that: (1) $A$ is not close to most of nearby examples around $D$, and $D$ is not close to most of nearby examples around $A$; (2) we used *Isomap* to maximally preserve the neighborhood information between any example and all other

Table 4.5: These two tables report performance comparison between $LP_{cosine}$ and $LP_{JS}$ and the results of three model selection criteria. In the lower table, we give out the mean values over 20 trials for three model selection criteria, $H(D)$, $H(W)$ and $H(Y_U)$. $\surd$ and $\times$ denote correct and wrong prediction results respectively, while $\circ$ means that any prediction is acceptable.

| | $LP_{cosine}$ vs. $LP_{JS}$ | |
|---|---|---|
| Data set | p-value | Significance |
| SENSEVAL-3 (10%) | 8.9e-5 | $\ll$ |
| SENSEVAL-3 (25%) | 9.0e-9 | $\ll$ |
| SENSEVAL-3 (50%) | 3.2e-10 | $\ll$ |
| SENSEVAL-3 (75%) | 7.7e-13 | $\ll$ |
| interest | 3.3e-2 | $>$ |
| line | 8.1e-2 | $\sim$ |

| | $H(D)$ | $H(W)$ | $H(Y_U)$ |
|---|---|---|---|
| Data set | cos. vs. JS | cos. vs. JS | cos. vs. JS |
| SENSEVAL-3 (10%) | 1.98e-1 1.94e-1 ($\surd$) | 2.01e-2 2.43e-2 ($\times$) | 3.96e-1 3.97e-1 ($\times$) |
| SENSEVAL-3 (25%) | 3.10e-1 2.88e-1 ($\surd$) | 2.04e-2 2.45e-2 ($\times$) | 3.29e-1 3.30e-1 ($\times$) |
| SENSEVAL-3 (50%) | 3.92e-1 3.47e-1 ($\surd$) | 2.03e-2 2.44e-2 ($\times$) | 2.8268e-1 2.8265e-1 ($\surd$) |
| SENSEVAL-3 (75%) | 4.40e-1 3.89e-1 ($\surd$) | 2.02e-2 2.43e-2 ($\times$) | 2.55e-1 2.54e-1 ($\surd$) |
| interest | 4.18e-1 4.20e-1 ($\surd$) | 3.66e-3 3.63e-3 ($\times$) | 5.34e-1 5.36e-1 ($\surd$) |
| line | 4.00e-1 3.97e-1 ($\circ$) | 1.94e-3 1.92e-3 ($\circ$) | 4.5216e-1 4.5157e-1 ($\circ$) |

example $D$ with label $\nabla$. But the correct label of $D$ should be + as shown in Figure 4.3 (b).

From Figure 4.3 (e), we see that smoothed co-training tried to maintain the class distribution when classifying unlabeled data. But the class distribution in unlabeled data is not consistent with that in initial labeled data, since there are only two classes in unlabeled data. This may explain the poor performance of co-training on this dataset.

With LP algorithm, the label information of example $B$ can transit to $D$ through other unlabeled examples. Then example $A$ will compete with $B$ and other unlabeled examples around $D$ when determining the label of $D$. In other words, the labels 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. This is also the reason why the other two examples E and F are correctly classified by LP, but misclassified by bootstrapping.

### 4.3.7 Experiment 5: $LP_{cosine}$ vs. $LP_{JS}$

Table 4.5 summarizes the performance comparison between $LP_{cosine}$ and $LP_{JS}$ on three datasets. We see that $LP_{JS}$ performs significantly better than $LP_{cosine}$ on SENSEVAL-3

examples, which caused the loss of neighborhood information between a few example pairs for obtaining a globally optimal solution.

data, but worse than $LP_{cosine}$ on "interest" corpus, and their performance is comparable on "line" corpus. It seems that there is no one distance measure that can consistently perform better than the other on all the datasets. This observation motivates us to automatically select a distance measure that will boost the performance of LP on a given dataset.

Cross-validation on labeled data is not feasible, since in the setting of semi-supervised learning, there are very few labeled examples available. In [164, 165], they suggested a label entropy criterion $H(Y_U)$ for optimal estimation of parameter $\sigma$ used in their semi-supervised algorithm, where $Y$ was $(l+u) \times c$ label matrix learned by their semi-supervised algorithm. In their semi-supervised algorithm, the value of each entry in $Y_U$ is between 0 and 1. Small entropy indicates that the value of each entry in $Y_U$ is close to 0 or 1. This captures the intuition that good parameter should result in a confident labeling. $H(Y_U)$ is a semi-supervised model selection criterion, since both labeled and unlabeled data are used for learning $Y_U$.

Here the value of parameter $\sigma$ is fixed, and we use their label entropy criterion to identify the optimal distance measure. Besides $H(Y_U)$, we suggest the other two criteria, supervised model selection criterion $H(D)$ and unsupervised model selection criterion $H(W)$.

$H(D)$ is the entropy of $c \times c$ distance matrix $D$ calculated on labeled data, where $D_{i,j}$ represents the average distance (cosine distance or JS distance) between examples from the $i$-th class and ones from the $j$-th class. If $i$ is equal to $j$, then $D_{i,j}$ is an intra-class distance, otherwise it is an inter-class distance. It is noted that $0 \leq D_{i,j} \leq 1$. Optimal distance measure usually increases the cohesion between the examples within the same class and the separability between the examples from difference classes at the same time, which results in small value of $D_{i,i}$ (close to 0) and large value of $D_{i,j}$ $(i \neq j)$ (close to 1), further, small entropy of matrix $D$. This is the intuition behind criterion $H(D)$.

Dash et al. (2000) introduced an unsupervised feature ranking criterion that measures the entropy of a distance matrix in which each entry $(i, j)$ represents the distance between the $i$-th and $j$-th examples. In the LP algorithm, $W$ may be considered as a distance matrix. Therefore, we used the entropy of $W$, $H(W)$, as model selection criterion. Optimal distance measure should result in clear cluster structure for dataset representation. Therefore the examples within the same cluster will be close to each other and those from different clusters will be more separable, which will result in small value of $H(W)$.

Let $Q$ be a $M \times N$ matrix. Function $H(Q)$ can measure the average entropy of matrix $Q$ by

$$H(Q) = -\frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (Q_{i,j} \log_2 Q_{i,j} + (1 - Q_{i,j}) \log_2 (1 - Q_{i,j})), \tag{4.5}$$

Specifically, on SENSEVAL-3 data, we calculated average scores of the three criteria in each trial by $\sum_w \frac{N_{w,labeled}}{\sum_w N_{w,labeled}} H(D_w)$, $\sum_w \frac{N_{w,all}}{\sum_w N_{w,all}} H(W_w)$, and $\sum_w \frac{N_{w,unlabeled}}{\sum_w N_{w,unlabeled}} H(Y_{U,w})$, where $D_w$, $W_w$, and $Y_{U,w}$ are the matrices constructed for each word $w$, and $N_{w,labeled}$, $N_{w,unlabeled}$ and $N_{w,all}$ are the number of labeled examples, unlabeled examples, and all the examples (labeled examples+unlabeled examples) of word $w$.

The optimal distance measure can be automatically selected by minimizing the average score of $H(D)$, $H(W)$ or $H(Y_U)$ over k (we set k=20 here) trials. Table 4.5 reports the automatic prediction results of these three criteria.

From Table 4.5, we can see that on SENSEVAL-3 data, $H(D)$ consistently predicted the optimal distance measure, but $H(W)$ failed on all the cases, and $H(Y_U)$ failed only when very few labeled examples were available (percentage of labeled data $\leq 25\%$). On "interest" corpus, $H(D)$ and $H(Y_U)$ selected the optimal distance measure, but $H(W)$ failed to do so.

Lower value of $H(D)$ implies larger inter-class distance and smaller intra-class distance in labeled data, which means clearer class structure. This will help LP algorithm to locate the correct class boundaries, which may interpret why $H(D)$ can consistently identify the optimal distance measure over all the cases to boost the performance of LP.

The poor performance of $H(W)$ may be due to the fact that important label information is not used when measuring the separability of examples in a dataset. The clearer cluster structure with the lower value of $H(W)$ may not be accordant with the true class structure in the data, which would deteriorate the performance of LP.

When very few labeled examples are available, the noise in labeled data makes it difficult to learn the classification boundary. Therefore the confidence of labeling may not properly predict the performance of learning algorithm with small labeled data, e.g., when the percentage of labeled data $\leq 25\%$ on SENSEVAL-3 data.

## 4.4 Summary

Several methods have been proposed to exploit bilingual resources, e.g., aligned parallel corpora, untagged monolingual corpora in two languages. The intuition behind these methods is that if different senses of an ambiguous word in source language are translated into different words in target language, then the translated words in the target language can serve as the tags of senses of this ambiguous word.

Another research line is to automatically generate monolingual sense tagged corpus without reference to the second language corpora. Bootstrapping is such a general scheme for minimizing the requirement of manually tagged corpus, which was proposed for sense disambiguation [51].

Compared to the bilingual corpora based approaches, the graph based method in our work does not need external resources, e.g., bilingual lexicons, aligned parallel corpora, and machine translation tools. This LP based sense disambiguation method benefits from initial tagged corpus and raw untagged monolingual corpus that can be cheaply acquired. In contrast with the bootstrapping technique, LP algorithm can utilize the cluster structure in unlabeled data by the use of graph structure to smooth the label of each unlabeled example.

In this chapter, we investigated a label propagation based semi-supervised learning algorithm for WSD. In its learning process, the labels of unlabeled examples are determined not only by nearby labeled examples, but also by nearby unlabeled examples. Experimental results demonstrated the potential of this graph based algorithm. It achieves better performance than SVM when only very few sense tagged examples are available. Moreover, its performance is also better than bootstrapping, co-training and their variants using majority voting, and comparable to bilingual bootstrapping. Finally we suggested an entropy based method to automatically identify a distance measure that can boost the performance of LP algorithm on a given dataset.

Semi-supervised sense disambiguation methods require tagged examples for each possible

sense of a target word. If there are no tagged instances for a sense (e.g., an undefined domain specific sense) in training data and there is a large amount of untagged corpora that contain instances for both general senses and the missed sense, then the sense tagger built on incomplete sense tagged corpus will mis-tag the instances of the missed sense. It is a problem encountered by traditional supervised or semi-supervised sense disambiguation methods. In next chapter, we will use partially supervised learning to address this problem by identifying a set of reliable sense tagged examples from untagged corpus for the missed sense if this missed sense occurs in the untagged corpus, and then building a sense tagger with the learned sense tagged data.

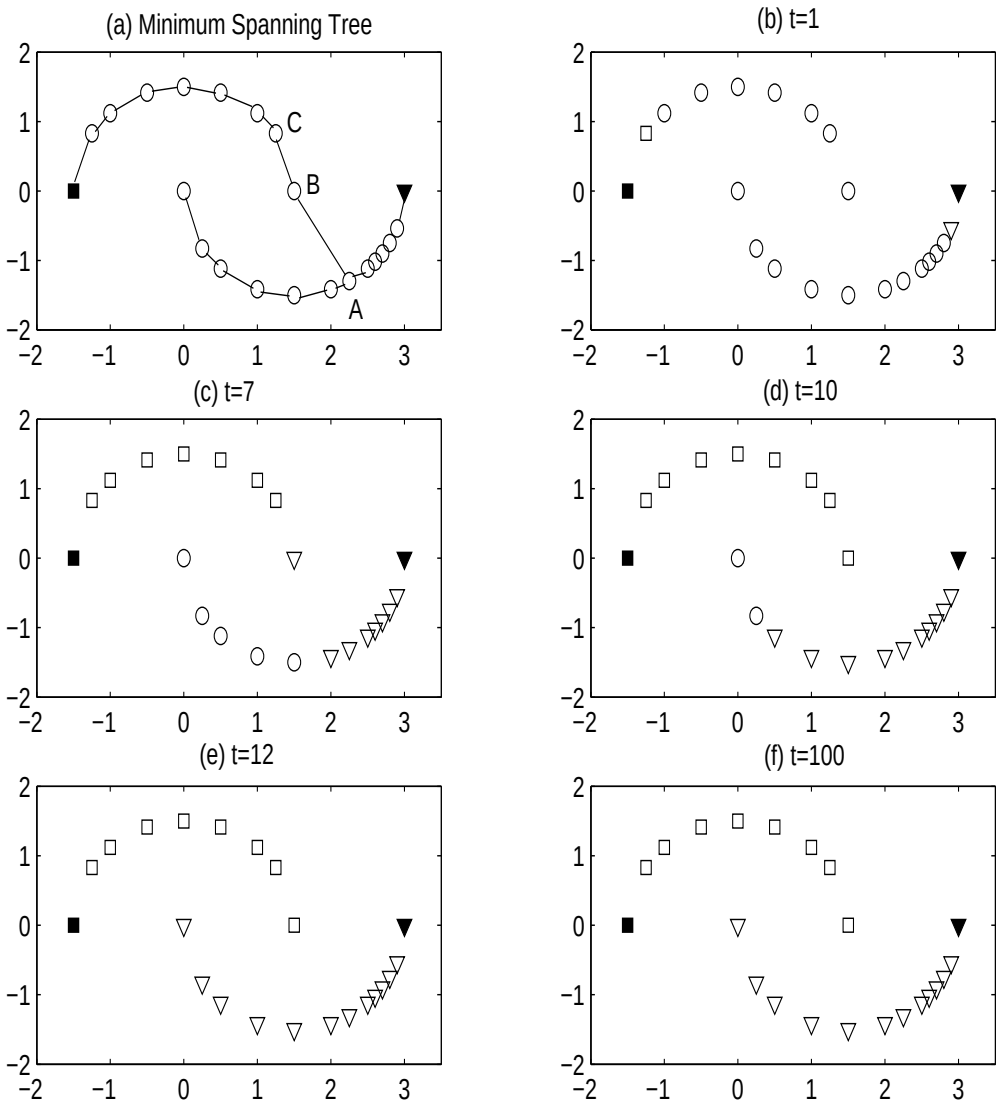


Figure 4.2: Classification result of LP on a two-moon pattern dataset. (a) Minimum spanning tree of this dataset. The convergence process of LP algorithm with $t$ varying from 1 to 100 is shown from (b) to (f).

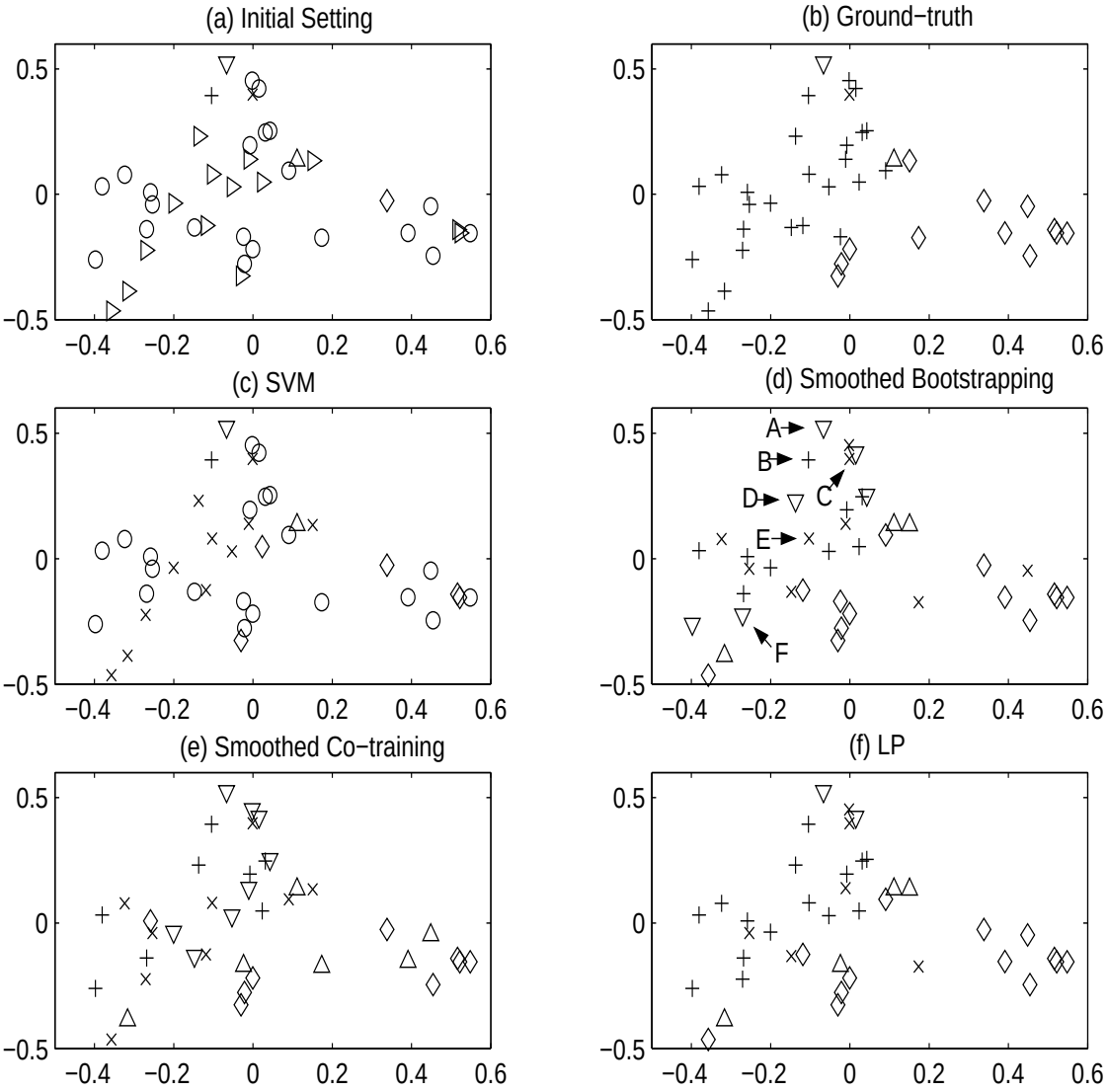


Figure 4.3: Comparison of sense disambiguation results between SVM, monolingual bootstrapping and LP on the data of the word "use". (a) only one labeled example for each sense of word "use" as training data before sense disambiguation ($\circ$ and $\triangleright$ denote the unlabeled examples in SENSEVAL-3 training set and test set respectively, and other five symbols ($+$, $\times$, $\triangle$, $\diamond$, and $\nabla$) represent the labeled examples with different sense tags sampled from SENSEVAL-3 training set.), (b) ground-truth result, (c) classification result by SVM (accuracy= $\frac{3}{14} = 21.4\%$), (d) classification result by bootstrapping, (accuracy= $\frac{6}{14} = 42.9\%$ ), (e) classification result by co-training (accuracy= $\frac{5}{14} = 35.7\%$ ) (f) classification result by LP (accuracy= $\frac{9}{14} = 64.3\%$ ).

# Chapter 5

# Partially Supervised Sense Disambiguation by Learning Sense Number from Tagged and Untagged Corpora

Previous supervised and semi-supervised sense disambiguation methods always rely on hand-crafted lexicon as sense inventories. But these resources may miss domain-specific senses, which lead to incomplete sense tagged corpus [1]. Therefore, sense taggers trained on incomplete tagged corpus will mistag the instances with senses undefined in sense inventories.

In this chapter we study the problem of partially supervised sense disambiguation with incomplete tagged corpus. Specifically, given incomplete sense-tagged examples and a large amount of untagged examples for a target word, we are interested in (1) labeling the instances of the target word in untagged corpus with sense tags occurring in the tagged corpus; (2) finding undefined senses of the target word from the untagged corpus if they occur in the untagged corpus, which will be represented by instances from the untagged corpus.

We propose an automatic method to estimate the sense number of a target word in mixed data (tagged corpus+untagged corpus) by maximizing a stability criterion defined on classification results over all the possible values of sense number. If the estimated sense number in the mixed data is equal to the sense number of the target word in tagged corpus, then there is no new sense in untagged corpus. Otherwise new senses will be represented by groups in which there is no instance from the tagged corpus. The stability criterion assesses the agreement between classification results on full mixed data and sampled mixed data. A partially supervised learning algorithm is used to classify mixed data into a given number of classes before stability evaluation.

Moreover, we empirically compare this algorithm with other related algorithms, e.g., SVM, one-class partially supervised classification algorithm [80], and clustering based partially supervised sense disambiguation algorithm.

This chapter is organized as follows. First, a partially supervised classification algorithm

[1] "incomplete sense tagged corpus" means that the sense tagged corpus does not include the instances of some senses for a target word, while these senses may occur in untagged corpus or test corpus.

is presented in section 5.1. Then in section 5.2 we use an example to show how this sense disambiguation algorithm works. Section 5.3 provides experimental results of this sense disambiguation algorithm on widely used benchmark corpora. Finally we conclude this work and suggest possible improvements in section 5.4.

## 5.1 Model Order Identification for Partially Supervised Classification

We perform partially supervised classification by following steps:

(1) Estimate the optimal value of model order of mixed data by maximizing a stability criterion defined on classification results from an extended label propagation algorithm (to be presented in section 5.1.1) over all possible values of model order in mixed data. The stability criterion assesses the agreement between classification result on full mixed data and that on sampled mixed data. An extended label propagation algorithm is used to classify the full or sampled mixed data into a given number of clusters before the stability assessment. We will provide the details of the model order identification procedure and the stability criterion in section 5.1.2.

(2) After model order identification (or cluster number estimation), we can obtain a partitioning of mixed data with the estimated number of clusters using the extended label propagation algorithm, where each cluster consists of similar examples from mixed data, and it satisfies two constraints: labeled examples with the same class label will stay in the same cluster, labeled examples with different class labels will stay in different clusters. In fact, this classification process has been performed on mixed data in the order identification procedure.

### 5.1.1 An Extended Label Propagation Algorithm

Let $X_{L+U} = \{x_i\}_{i=1}^{n}$ be a set of labeled and unlabeled examples (mixed data) for a target word, where $x_i$ represents the $i$-th example, and $n$ is the total number of examples. Let $S_L = \{s_j\}_{j=1}^{c}$ denote the class label set in $X_L$, where $X_L$ consists of the first $l$ examples $x_g (1 \leq g \leq l)$ that are labeled as $y_g$ ($y_g \in S_L$). Let $X_U$ denote other $u$ ($l + u = n$) examples $x_h (l + 1 \leq h \leq n)$ that are unlabeled.

Let $Y^0_{X_{L+U}} \in N^{|X_{L+U}| \times |S_L|}$ represent initial soft labels attached to labeled examples, where $Y^0_{X_{L+U},ij} = 1$ if $y_i$ is $s_j$ and 0 otherwise. Let $Y^0_{X_L}$ be the top $l$ rows of $Y^0_{X_{L+U}}$ and $Y^0_{X_U}$ be the remaining $u$ rows. $Y^0_{X_L}$ is consistent with the labeling in labeled data, and the initialization of $Y^0_{X_U}$ can be arbitrary.

Let $k$ denote the possible value of model order in mixed data $X_{L+U}$, and $k_{X_L}$ be the number of classes in initial tagged data $X_L$. Note that $k_{X_L} = |S_L|$, and $k \geq k_{X_L}$.

The classification algorithm in the order identification process should be able to accept labeled data $D_L$ [2], unlabeled data $D_U$ [3] and model order $k$ as input, and assign a class label or a cluster index to each instance in $D_U$ as output. Traditional supervised or semi-supervised

[2] $D_L$ may be the dataset $X_L$ or a subset sampled from $X_L$.

[3] $D_U$ may be the dataset $X_U$ or a subset sampled from $X_U$.

Table 5.1: An extended label propagation algorithm.

| | |
|---|---|
| | Function: ELP($D_L$, $D_U$, $k$, $Y^0_{D_L+D_U}$) |
| | Input: labeled examples $D_L$, unlabeled examples $D_U$, model order $k$, initial labeling matrix $Y^0_{D_L+D_U}$; |
| | Output: the labeling matrix $Y_{D_U}$ for $D_U$; |
| 1 | If $k < k_{X_L}$ then |
| | $Y_{D_U}$=NULL; |
| 2 | Else if $k = k_{X_L}$ then |
| | Run plain label propagation algorithm on $D_U$ with $Y_{D_U}$ as output; |
| 3 | Else then |
| 3.1 | Estimate the size of tagged data set of each new class; |
| 3.2 | Generate tagged examples from $D_U$ for $(k_{X_L}+1)$-th to $k$-th new classes; |
| 3.3 | Run the plain label propagation algorithm on $D_U$ with augmented tagged dataset as labeled data; |
| 3.4 | $Y_{D_U}$ is the output from plain label propagation algorithm; |
| | End if |
| 4 | Return $Y_{D_U}$; |

algorithms (e.g. SVM, label propagation algorithm [164]) cannot classify the examples in $D_U$ into $k$ clusters if $k > k_{X_L}$. The semi-supervised k-means clustering algorithm [146] may be used to perform clustering analysis on mixed data, but its efficiency is a problem for clustering analysis on a very large dataset since multiple restarts are usually required to avoid local optima and multiple iterations will be run in each clustering process for optimizing a clustering solution.

In this work, we propose an alternative method, an extended label propagation algorithm (ELP), which can classify the examples in $D_U$ into $k$ clusters. If the value of $k$ is equal to $k_{X_L}$, then ELP is identical with the plain label propagation algorithm (LP) [164]. Otherwise, if the value of $k$ is greater than $k_{X_L}$, we will perform classification by following steps:

(1) estimate the size of the dataset of each new class as $size_{new\_class}$ by identifying the examples of new classes using the "Spy" technique [4] and assuming that new classes are equally distributed;

(2) $D_L^{'} = D_L$, $D_U^{'} = D_U$;

(3) remove tagged examples of the $m$-th new class ($k_{X_L} + 1 \leq m \leq k$) from $D_L^{'}$ [5] and train a classifier on this labeled dataset without the $m$-th class;

(4) the classifier is then used to classify the examples in $D_U^{'}$;

(5) the least confidently unlabeled point $x_{class\_m} \in D_U^{'}$, together with its label $m$, is added to the labeled data $D_L^{'} = D_L^{'} + x_{class\_m}$, and $D_U^{'} = D_U^{'} - x_{class\_m}$;

[4]The "Spy" technique was proposed in [80]. Our re-implementation of this technique consists of three steps: (1) sample a small subset $D_L^s$ with the size $15\% \times |D_L|$ from $D_L$; (2) train a classifier with tagged data $D_L - D_L^s$; (3) classify $D_U$ and $D_L^s$, and then select some examples from $D_U$ as the dataset of new classes, which have the classification confidence less than the average of that in $D_L^s$. Classification confidence of the example $x_i$ is defined as the absolute value of the difference between two maximum values from the $i$-th row in labeling matrix.

[5]Initially there are no tagged examples for the $m$-th class in $D_L^{'}$. Therefore we do not need to remove tagged examples for this new class, and then directly train a classifier with $D_L^{'}$.

(6) steps (3) to (5) are repeated for each new class till the augmented tagged data set is large enough (here we try to select $size_{new\_class}/4$ examples with their sense tags as tagged data for each new class);

(7) use the plain LP algorithm to classify remaining unlabeled data $D'_U$ with $D'_L$ as labeled data.

Table 5.1 shows the details of this extended label propagation algorithm.

Next we will describe the plain label propagation algorithm.

Define $W_{ij} = exp(-\frac{d_{ij}^2}{\sigma^2})$ if $i \neq j$ and $W_{ii} = 0$ $(1 \leq i, j \leq |D_L + D_U|)$, where $d_{ij}$ is the distance (e.g., Euclidean distance) between the example $x_i$ and $x_j$, and $\sigma$ is used to control the weight $W_{ij}$.

Define $|D_L + D_U| \times |D_L + D_U|$ 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 $\overline{T}_{ij} = T_{ij} / \sum_{k=1}^{n} T_{ik}$.

The classification solution is obtained by $Y_{D_U} = (I - \overline{T}_{uu})^{-1} \overline{T}_{ul} Y^0_{D_L}$. $I$ is $|D_U| \times |D_U|$ identity matrix. $\overline{T}_{uu}$ and $\overline{T}_{ul}$ are acquired by splitting matrix $\overline{T}$ after the $|D_L|$-th row and the $|D_L|$-th column into 4 sub-matrices.

### 5.1.2 Model Order Identification Procedure

For achieving the model order identification ability, we use a cluster validation based criterion [73] to infer the optimal value of model order of $X_{L+U}$.

The model order identification procedure can be formulated as:

$$\hat{k}_{X_{L+U}} = argmax_{K_{min} \leq k \leq K_{max}} \{CV(X_{L+U}, k, q, Y^0_{X_{L+U}})\}. \tag{5.1}$$

$\hat{k}_{X_{L+U}}$ is the estimated model order in $X_{L+U}$, $K_{min}$ (or $K_{max}$) is the minimum (or maximum) value of model order, and $k$ is the possible value of model order in $X_{L+U}$. Note that $k \geq k_{X_L}$. We set $K_{min} = k_{X_L}$. $K_{max}$ will be set as a value greater than any possible ground-truth value. $CV$ is a cluster validation based evaluation function, or stability criterion. Table 5.2 shows the details of this function. We set $q$, the resampling frequency for estimation of stability score, as 20. $\alpha$ is set as 0.90. The random predictor assigns uniformly distributed class labels to each instance in a given dataset. We run this $CV$ procedure for each value of $k$. The value of $k$ that maximizes this $CV$ function will be selected as the estimation of model order. At the same time, we can obtain a partitioning of $X_{L+U}$ with $\hat{k}_{X_{L+U}}$ clusters.

The function $M(C^\mu, C)$ in Table 5.2 is given by [73]:

$$M(C^\mu, C) = \frac{\sum_{i,j} 1\{C^\mu_{i,j} = C_{i,j} = 1, x_i, x_j \in X^\mu_U\}}{\sum_{i,j} 1\{C_{i,j} = 1, x_i, x_j \in X^\mu_U\}}, \tag{5.2}$$

where $X^\mu_U$ is the untagged data in $X^\mu_{L+U}$, $X^\mu_{L+U}$ is a subset with the size $\alpha|X_{L+U}|$ $(0 < \alpha < 1)$ sampled from $X_{L+U}$, $C$ or $C^\mu$ is $|X_U| \times |X_U|$ or $|X^\mu_U| \times |X^\mu_U|$ connectivity matrix based on classification solutions computed on $X_U$ or $X^\mu_U$ respectively. The connectivity matrix $C$ is defined as: $C_{i,j} = 1$ if $x_i$ and $x_j$ belong to the same cluster, otherwise $C_{i,j} = 0$. $C^\mu$ is calculated in the same way.

$M(C^\mu, C)$ measures the proportion of example pairs in each cluster computed on $X_U$ that are also assigned into the same cluster by the classification solution on $X^\mu_U$. Clearly,

Table 5.2: Stability criterion.

| | |
|---|---|
| | Function: CV($X_{L+U}$, $k$, $q$, $Y^0_{X_{L+U}}$) |
| | Input: data set $X_{L+U}$, model order $k$, and sampling frequency $q$; |
| | Output: the score of the merit of $k$; |
| 1 | Run the extended label propagation algorithm with $X_L$, $X_U$, $k$ and $Y^0_{X_{L+U}}$; |
| 2 | Construct connectivity matrix $C_k$ based on above classification solution on $X_U$; |
| 3 | Use a random predictor $\rho_k$ to assign uniformly drawn labels to each vector in $X_U$; |
| 4 | Construct connectivity matrix $C_{\rho_k}$ based on above classification solution on $X_U$; |
| 5 | For $\mu = 1$ to $q$ do |
| 5.1 | Randomly sample a subset $X^\mu_{L+U}$ with the size $\alpha\|X_{L+U}\|$ from $X_{L+U}$, $0 < \alpha < 1$; |
| 5.2 | Run the extended label propagation algorithm with $X^\mu_L$, $X^\mu_U$, $k$ and $Y^{0\mu}$; |
| 5.3 | Construct connectivity matrix $C^\mu_k$ using above classification solution on $X^\mu_U$; |
| 5.4 | Use $\rho_k$ to assign uniformly drawn labels to each vector in $X^\mu_U$; |
| 5.5 | Construct connectivity matrix $C^\mu_{\rho_k}$ using above classification solution on $X^\mu_U$; |
| | Endfor |
| 6 | Evaluate the merit of $k$ using following formula: $M_k = \frac{1}{q}\sum_\mu (M(C^\mu_k, C_k) - M(C^\mu_{\rho_k}, C_{\rho_k}))$, where $M(C^\mu, C)$ is given by equation (2); |
| 7 | Return $M_k$; |

$0 \leq M \leq 1$. Intuitively, if the value of $k$ is identical with the true value of model order, then classification results on different subsets generated by sampling should be similar with that on full dataset. In the other words, the classification solution with the true model order as input is robust against resampling, which gives rise to a local optimum of $M(C^\mu, C)$.

In this algorithm, we normalize $M(C^\mu_k, C_k)$ by the equation in step 6 of Table 5.2, which makes our objective function different from the figure of merit (equation ( 5.2)) proposed in [73]. The reason to normalize $M(C^\mu_k, C_k)$ is that $M(C^\mu_k, C_k)$ tends to decrease when increasing the value of $k$ [65]. Therefore for avoiding the bias that the smaller value of $k$ is to be selected as the model order, we use the cluster validity of a random predictor to normalize $M(C^\mu_k, C_k)$.

If $\hat{k}_{X_{L+U}}$ is equal to $k_{X_L}$, then there is no new sense in $X_U$. Otherwise ($\hat{k}_{X_{L+U}} > k_{X_L}$) new senses may be represented by clusters in which there is no instance from $X_L$.

## 5.2 A Walk-Through Example

For understanding how this partially supervised sense disambiguation algorithm works, we use a dataset with four-disc pattern as an example, shown in Figure 5.1(a). Each disc

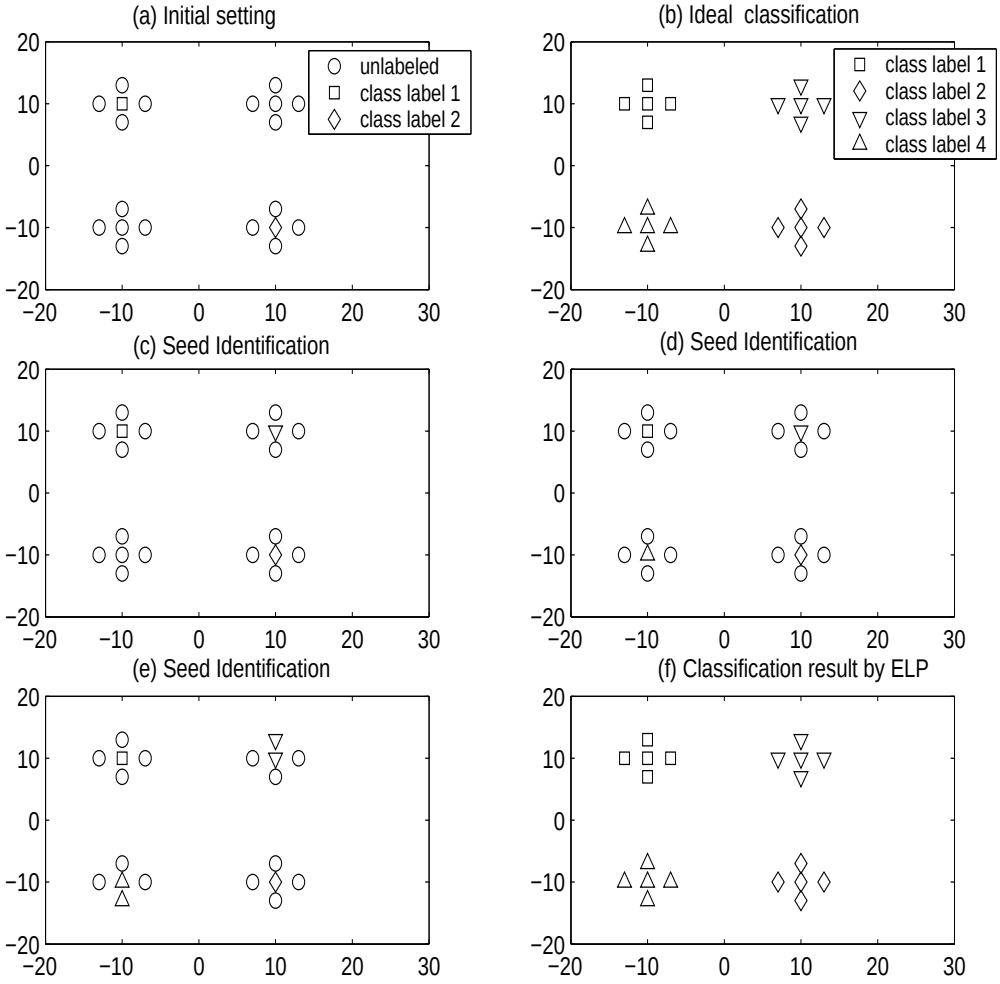


Figure 5.1: Classification process on a four-disc pattern dataset. (a) The four-disc pattern dataset with two labeled points; (b) ideal classification; (c) to (e) seed identification process in ELP ; (f) final classification result by ELP using initial labeled data and identified seeds as labeled data.

pattern group consists of 5 points, while only the centroid points in upper-left and lower-right groups are labeled. The distance metric is Euclidian distance. We can see that the points in one disc should be more similar to each other than the points across the discs, as shown in Figure 5.1(b).

We ran this model order identification process on this dataset and obtained final classification result shown in Figure 5.1(f). For model order identification, we set $K_{min} = k_L = 2$, $K_{max} = 6$. The scores of stability criterion for each value of model order were 0.08, 0.37, 0.49, 0.41, and 0.45. Then the estimated model order was 4, which is equal to the true value of class number.

Figure 5.1(c) to (e) show the classification process of ELP when model order was 4 in the order identification process. Firstly ELP estimated the size of seed set for each new class as $\tau \cdot \text{Class\_Size} = 0.25 \cdot 9 \approx 2$ ($\tau$ is a parameter to be specified manually). Then ELP identified the point with minimum value of $Sig$ criterion as a seed for one of the two

Table 5.3: Description of the percentage of official training data used as tagged data when the instances with different sense sets are removed from official training data.

| | The percentage of official training data used as tagged data |
|---|---|
| $S_{subset} = \{s_1\}$ | 42.8% |
| $S_{subset} = \{s_2\}$ | 76.7% |
| $S_{subset} = \{s_3\}$ | 89.1% |
| $S_{subset} = \{s_1, s_2\}$ | 19.6% |
| $S_{subset} = \{s_1, s_3\}$ | 32.0% |
| $S_{subset} = \{s_2, s_3\}$ | 65.9% |

new classes, as shown in Figure 5.1(c). After that, another point was selected as a seed for the other new class, as shown in Figure 5.1(d). Figure 5.1(e) shows the final seed sets of the two new classes. By the use of initial labeled data of two known classes and automatically generated labeled data of two unknown classes, ELP performed classification on the remaining unlabeled points. The final classification result is shown in Figure 5.1(f), which is identical with the ideal classification result.

## 5.3 Experiments and Results

### 5.3.1 Experiment Design

We evaluated the ELP based model order identification algorithm (implemented by ourselves) on the data in English lexical sample task of SENSEVAL-3 (including all the 57 English words ) [6] for WSD, and further empirically compared it with other state of the art classification methods, including SVM [7] (the state of the art method for supervised WSD[88]), a one-class partially supervised classification algorithm [80] [8], and a semi-supervised k-means clustering based model order identification algorithm (implemented by ourselves).

Given an incomplete tagged corpus for a target word, SVM does not have the ability to find the new senses from untagged corpus. Therefore it labels all the instances in the untagged corpus with sense tags from $S_L$.

Given a set of positive examples for a class and a set of unlabeled examples, the one-class partially supervised classification algorithm, LPU (Learning from Positive and Unlabeled examples) [80], learns a classifier in four steps:

Step 1: Identify a small set of reliable negative examples from unlabeled examples by the use of a classifier.

Step 2: Build a classifier using positive examples and automatically selected negative examples.

Step 3: Iteratively run previous two steps until no unlabeled examples are classified as negative ones or the unlabeled set is null.

Step 4: Select a good classifier from the set of classifiers constructed above.

[6] Available at http://www.senseval.org/senseval3

[7] we used a linear $SVM^{light}$, available at http://svmlight.joachims.org/.

[8] Available at http://www.cs.uic.edu/∼liub/LPU/LPU-download.html

For comparison, LPU [9] was run to perform classification on $X_U$ for each class in $X_L$. The label of each instance in $X_U$ was determined by maximizing the classification score from LPU output for each class. If the maximum score of an instance is negative, then this instance will be labeled as a new class. Note that LPU classifies $X_{L+U}$ into $k_{X_L}+1$ groups in most of cases.

The clustering based partially supervised sense disambiguation algorithm was implemented by replacing ELP with the semi-supervised k-means clustering algorithm [146] in the model order identification procedure. The label information in labeled data was used to guide the semi-supervised clustering on $X_{L+U}$. Firstly, the labeled data may be used to determine initial cluster centroids. If the cluster number is greater than $k_{X_L}$, the initial centroids of clusters for new classes will be assigned as randomly selected instances. Secondly, in the clustering process, the instances with the same class label will stay in the same cluster, while the instances with different class labels will belong to different clusters. For better clustering solution, this clustering process will be restarted three times. Clustering process will be terminated when clustering solution converges or the number of iteration steps is more than 30. $K_{min} = k_{X_L} = |S_L|$, $K_{max} = K_{min} + m$. $m$ is set as 4.

The data for English lexical samples task in SENSEVAL-3 consists of 7860 examples as official training data, and 3944 examples as official test data for 57 English words. The number of senses of each English word varies from 3 to 11.

We evaluated these four algorithms with different incomplete tagged datasets. Given official training data of word $w$, we constructed incomplete tagged data $X_L$ by removing the all the tagged instances from official training data that have sense tags from $S_{subset}$, where $S_{subset}$ is a subset of the ground-truth sense set $S$ for $w$, and $S$ consists of the sense tags in official training set for $w$. The removed training data and official test data of $w$ were used as $X_U$. Note that $S_L = S - S_{subset}$. Then we ran these four algorithm for each target word $w$ with $X_L$ as tagged data and $X_U$ as untagged data, and evaluated their performance using the accuracy on official test data of all the 57 words [10]. We conducted six experiments for each target word $w$ by setting $S_{subset}$ as $\{s_1\}$, $\{s_2\}$, $\{s_3\}$, $\{s_1, s_2\}$, $\{s_1, s_3\}$, or $\{s_2, s_3\}$, where $s_i$ is the i-th most frequent sense of $w$. $S_{subset}$ cannot be set as $\{s_4\}$ since some words have only three senses. Table 5.3 lists the percentage of official training data used as tagged data (the number of examples in incomplete tagged data divided by the number of examples in official training data) when we removed the instances with sense tags from $S_{subset}$ for all the 57 words. If $S_{subset} = \{s_3\}$, then most of sense tagged examples still stay in tagged data. If $S_{subset} = \{s_1, s_2\}$, then there are very few tagged examples in tagged data. If no instances are removed from official training data, then the value of percentage is 100%.

We used Jensen-Shannon (JS) divergence [78] as distance measure for semi-supervised clustering and ELP, since plain LP with JS divergence achieves better performance than that with cosine similarity on SENSEVAL-3 data [104].

For the plain LP algorithm in ELP, we constructed connected graphs as follows: two instances $u, v$ will be connected by an edge if $u$ is among $v$'s 10 nearest neighbors, or if $v$

[9] The three parameters in LPU were set as follows: "-s1 spy -s2 svm -c 1". It means that we used the spy technique for step 1 in LPU, the SVM algorithm for step 2, and selected the first or the last classifier as the final classifier. It is identical with the algorithm "Spy+SVM IS" in Liu et al. (2003).

[10] Here the accuracy is as same as the precision measure in SENSEVAL-3, and the recall of all the methods we evaluated is 100% since we attempted to label all the instances in test data

is among $u$'s 10 nearest neighbors as measured by cosine or JS distance measure (following [164]).

We used three types of features to capture the information in all the contextual sentences of target words in SENSEVAL-3 data for all the four algorithms: part-of-speech of neighboring words with position information, words in topical context without position information (after removing stop words), and local collocations (as same as the feature set used in [69] except that we did not use syntactic relations). We removed the features with occurrence frequency (counted in both official training set and official test set) less than 3 times.

If the estimated number of senses (or sense number) is more than the number of senses in the initial tagged corpus $X_L$, then the results from order identification based methods will consist of the instances from new classes. When assessing the agreement between these classification results and the ground-truth results on official test set, we will encounter the problem that there is no sense tag for each instance in new classes. Slonim and Tishby (2000) proposed to assign documents in each cluster with the most dominant class label in that cluster, and then conducted evaluation on these labeled documents. Here we will follow their method for assigning sense tags to new classes from LPU, clustering based order identification process, and ELP based order identification process. We assigned the instances from new classes with the dominant sense tag in that cluster. The result from LPU always includes only one new class. We assigned the instances from the new class with the dominant sense tag in that cluster. When all instances have their sense tags, we evaluated the results using the accuracy on official test set in SENSEVAL-3.

### 5.3.2 Results on Sense Disambiguation

Table 5.4 summarizes the accuracy of SVM, LPU, the semi-supervised k-means clustering algorithm with correct sense number $|S|$ or estimated sense number $\hat{k}_{X_{L+U}}$ as input, and the ELP algorithm with correct sense number $|S|$ or estimated sense number $\hat{k}_{X_{L+U}}$ as input using various incomplete tagged data. The bottom row in Table 5.4 lists the average accuracy of each algorithm over six experimental settings. Using $|S|$ as input means that we do not perform order identification procedure, while using $\hat{k}_{X_{L+U}}$ as input is to perform order identification and obtain the classification results on $X_U$ at the same time.

Given the correct sense number as input, the average accuracy of the ELP algorithm and the clustering algorithm are 52.5% and 46.1% respectively. When using the estimated sense number as input, the ELP algorithm and the clustering algorithm achieved 48.9% and 43.8% as the average accuracy. We can see that the ELP based method outperforms the clustering based method in terms of average accuracy under the same experiment setting, e.g., using the correct sense number as input or the estimated sense number as input. Moreover, using the correct sense number as input helps to improve the overall performance of both clustering based method and ELP based method. The two methods, the ELP based method and the clustering based method, outperform the other two algorithms, SVM and LPU.

Comparing the performance of the same system with different tagged datasets (from the first experiment to the third experiment, and from the fourth experiment to the sixth experiment), we can see that in most of cases, the performance of SVM, LPU, the ELP based method, and the clustering based method was improved when using more labeled data. For example, when $S_{subset} = s_1$ (the most frequent sense is missing), the accuracy of SVM, LPU,

Table 5.4: This table summarizes the accuracy of SVM, LPU, the semi-supervised k-means clustering algorithm with correct sense number $|S|$ or estimated sense number $\hat{k}_{X_{L+U}}$ as input, and the ELP algorithm with correct sense number $|S|$ or estimated sense number $\hat{k}_{X_{L+U}}$ as input on official test data of English Lexical Sample task in SENSEVAL-3 when given various incomplete tagged datasets.

| | SVM | LPU | The clustering algorithm with $\|S\|$ as input | The ELP algorithm with $\|S\|$ as input | The clustering algorithm with $\hat{k}_{X_{L+U}}$ as input | The ELP algorithm with $\hat{k}_{X_{L+U}}$ as input |
|---|---|---|---|---|---|---|
| $S_{subset} = \{s_1\}$ | 30.6% | 22.3% | 43.9% | 47.8% | 40.0% | 38.7% |
| $S_{subset} = \{s_2\}$ | 59.7% | 54.6% | 44.0% | 62.4% | 48.5% | 62.6% |
| $S_{subset} = \{s_3\}$ | 67.0% | 53.4% | 48.7% | 67.2% | 52.4% | 69.1% |
| $S_{subset} = \{s_1, s_2\}$ | 14.6% | 13.1% | 44.4% | 40.2% | 35.6% | 33.0% |
| $S_{subset} = \{s_1, s_3\}$ | 25.7% | 21.1% | 48.5% | 37.9% | 39.8% | 31.0% |
| $S_{subset} = \{s_2, s_3\}$ | 56.2% | 53.1% | 47.3% | 59.4% | 46.6% | 58.7% |
| Average accuracy | 42.3% | 36.3% | 46.1% | 52.5% | 43.8% | 48.9% |

the clustering based method with $|S|$ as input, and the ELP based method with $|S|$ as input, the clustering based method with $\hat{k}_{X_{L+U}}$ as input, and the ELP method with $\hat{k}_{X_{L+U}}$ as input are 30.6%, 22.3%, 43.9%, 47.8%, 40.0%, and 38.7%. The performance of these methods were improved to 67.0%, 53.4%, 48.7%, 67.2%, 52.4%, and 69.1% when $S_{subset} = s_3$ (only a rare sense is missing). Furthermore, ELP based method outperforms other methods in terms of accuracy when rare senses (e.g. $s_3$) are missing in the tagged data. It seems that the ELP based method has the ability to find rare senses with the use of tagged and untagged corpora.

The LPU algorithm can deal with only one-class classification problem. Therefore the labeled data of other classes cannot be used when determining the positive labeled data for current class. The ELP algorithm can use the labeled data of all the known classes to determine the seeds of new classes. It may explain why LPU does not outperform ELP although LPU can correctly estimate the sense number in $X_{L+U}$ when only one sense is missing in $X_L$.

When only few labeled examples are available, the noise in labeled data makes it difficult to learn confidence score (each entry in $Y_{D_U}$). Therefore using the classification confidence criterion may lead to poor performance of seed selection for new classes if confidence score is not accurate. It may explain why ELP based method does not outperform clustering based method with small labeled data (e.g., $S_{subset} = \{s_1\}$).

Table 5.5: These two tables provide the mean and standard deviation of absolute values of the difference between ground-truth results $|S|$ and sense numbers estimated by clustering or ELP based order identification procedure respectively.

| | The clustering based method | The ELP based method |
|---|---|---|
| $S_{subset} = \{s_1\}$ | 1.3±1.1 | 2.2±1.1 |
| $S_{subset} = \{s_2\}$ | 2.4±0.9 | 2.4±0.9 |
| $S_{subset} = \{s_3\}$ | 2.6±0.7 | 2.6±0.7 |
| $S_{subset} = \{s_1, s_2\}$ | 1.2±0.6 | 1.6±0.5 |
| $S_{subset} = \{s_1, s_3\}$ | 1.4±0.6 | 1.8±0.4 |
| $S_{subset} = \{s_2, s_3\}$ | 1.8±0.5 | 1.8±0.5 |

### 5.3.3 Results on Sense Number Estimation

Table 5.5 provides the mean and standard deviation of absolute difference values between ground-truth results $|S|$ and sense numbers estimated by the clustering or ELP based order identification procedures respectively. For example, if the ground truth sense number of word $w$ is $k_w$, and the estimated value is $\hat{k}_w$, then the absolute value of the difference between these two values is $|k_w - \hat{k}_w|$. Therefore we can have this value for each word. Then we calculated the mean and deviation on this array of absolute values. LPU does not have the order identification capability since it always assumes that there is one new class in unlabeled data, and does not further differentiate the instances from these new classes. Therefore we do not provide the order identification results of LPU.

From the results in Table 5.5, we can see that the estimated sense numbers are closer to the ground truth results when using less labeled data. Moreover, the clustering based method performs better than the ELP based method in terms of order identification when using small labeled data (e.g., $S_{subset} = \{s_1\}$). It seems that ELP is not robust to the noise in small labeled data, compared with the semi-supervised k-means clustering algorithm.

## 5.4 Summary

Here we presented a model order identification based partially supervised classification algorithm, which can classify unlabeled data into positive and negative examples, and further group negative examples into a natural number of clusters. Experimental results on SENSEVAL-3 data for word sense disambiguation task indicate that this classification process with ELP as the base classifier achieves better performance than SVM, LPU, and the classification process with the semi-supervised k-means clustering as the base classifier.

The work closest to ours is partially supervised classification or building classifiers using positive and unlabeled examples, which has been studied in machine learning community [36, 80, 81, 160]. They try to learn from positive and unlabeled examples for a given class.

After the learning procedure, unlabeled examples will be classified into positive examples and negative examples. These methods always assume the occurrence of negative examples in unlabeled data. Moreover, they cannot group negative examples into meaningful clusters. In contrast, our algorithm can find the occurrences of new senses (represented by negative examples) and further differentiate the new senses by grouping those negative examples into clusters. Semi-supervised clustering [146] may be used to perform classification by the use of labeled and unlabeled examples, but it encounters the same problem of partially supervised classification that model order cannot be automatically estimated.

Levine and Domany (2001) and Lange et al. (2002) proposed cluster validation based criteria for cluster number estimation. However, they showed the application of the cluster validation method only for unsupervised learning. Our work can be considered as an extension of their methods in the setting of partially supervised learning by incorporating a partially supervised classification algorithm (the ELP algorithm) in the order identification process.

In natural language processing community, the work that is closely related to ours is word sense discrimination which can induce senses by grouping occurrences of a word into clusters without the use of labeled training data and sense inventories [46, 108, 126].

Schutze's approach firstly selected important contextual words using $\chi^2$ or local frequency criterion. With the $\chi^2$ based criterion, those contextual words whose occurrence depended on whether the ambiguous word occurred were chosen as features. When using local frequency criterion, his algorithm selected top $n$ most frequent contextual words as features. Then each context of occurrences of a target word was represented by second order co-occurrence based context vector. Singular value decomposition (SVD) was conducted to reduce the dimensionality of context vectors. Then the reduced context vectors were grouped into a pre-defined number of clusters whose centroids corresponded to senses of target word.

Pedersen and Bruce (1997) described an experimental comparison of three clustering algorithms for word sense discrimination. Their feature sets included morphology of a target word, part of speech of contextual words, absence or presence of particular contextual words, and collocation of frequent words. Then occurrences of target word were grouped into a pre-defined number of clusters. Similar with many other algorithms, their algorithm also required the cluster number to be provided.

Fukumoto and Suzuki (1999) presented a term weight learning algorithm for verb sense disambiguation, which can automatically extract nouns co-occurring with verbs and identify the number of senses of an ambiguous verb. The weakness of their method is to assume that nouns co-occurring with verbs are disambiguated in advance and the number of senses of target verb is no less than two.

Another related work is unknown word sense detection by Erk (2006). He addressed the problem of unknown word sense detection as the identification of corpus occurrences that are not covered by a given sense inventory. He modeled this problem as an instance of outlier detection, using a simple nearest neighbor-based approach to measure the resemblance of a new item to a training set. His work has the problem that occurrences with unknown senses cannot be grouped into meaningful clusters even if there are two or more unknown senses in test data.

Word sense discrimination methods use unsupervised methods to solve sense disambiguation problem without the use of labeled training data, while our algorithm can utilize both

labeled data and unlabeled data. In comparison with semi-supervised sense disambiguation methods, e.g., bootstrapping [155], our algorithm can find missing senses from unlabeled data.

# Chapter 6

# Conclusion

This thesis reports the results of our study on using unsupervised learning, semi-supervised learning, and partially supervised learning to address various problems encountered by traditional sense disambiguation methods.

Previous unsupervised sense discrimination methods usually require the sense number of a target word as input for clustering analysis. We used a GMM+MDL based clustering method to address this problem by automatically estimating the sense number of the target word from untagged corpus.

Previous semi-supervised sense disambiguation methods, e.g., bootstrapping, cannot use the cluster structure in unlabeled data to help learn the labeling function. Here we investigated a more principled graph based semi-supervised learning method, the label propagation algorithm, for sense disambiguation. It represented labeled and unlabeled examples and their distances as the nodes and the weights of edges of a graph, and tried to obtain a labeling function to satisfy two constraints: 1) it should be fixed on the labeled nodes, 2) it should be smooth on the whole graph.

Previous supervised and semi-supervised sense disambiguation methods require sense tagged examples for each possible sense of a target word. If there are no tagged instances for one sense (e.g., a domain specific sense) in training data and there is a large amount of untagged corpora that contain instances for both general senses and the missed sense, then the sense tagger built on the incomplete sense tagged corpus will mis-tag the instances of the missed sense in the untagged corpora. We presented a partially supervised sense disambiguation algorithm to address this problem by identifying a set of reliable sense tagged examples from the untagged corpus for the missed sense if this sense occurs in the untagged corpus, and then building a sense tagger with the learned sense tagged data.

Next, we will provide the details of our work.

## 6.1 Word Sense Discrimination with Feature Selection and Order Identification Capabilities

We presented a word sense discrimination algorithm with feature selection and order identification capabilities. Our approach to sense discrimination improved previous work by

incorporating constrained feature selection and sense number estimation into sense discrimination procedure. Feature selection was formalized as a constrained optimization problem, and then the output of this procedure was a set of important features to determine word senses. Furthermore, our work extended the feature selection procedure backward to the construction of feature vectors. Both cluster structure and cluster number were estimated by minimizing a MDL criterion.

Experimental results showed that this algorithm can retrieve important features, estimate cluster number automatically, and achieve better performance in terms of average accuracy than Schutze's $CGD$ algorithm which requires cluster number as input.

## 6.2 Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning

In this work, we investigated a label propagation based semi-supervised learning algorithm for WSD, which used the graph structure to smooth the labeling function.

The experimental results demonstrated the potential of this graph based algorithm. It achieves better performance than SVM when only very few sense tagged examples are available. Moreover, its performance is also better than bootstrapping, co-training and their variants using majority voting, and comparable to bilingual bootstrapping.

Supervised learning algorithms cannot utilize untagged corpora for sense disambiguation, but LP algorithm can combine it in the learning process. Small labeled data cannot reveal the structure in each class, which will deteriorate the performance of supervised learning algorithms or local consistency based semi-supervised learning algorithms. LP can use unlabeled data to find the class structure, which helps to improve its performance.

Finally, we suggested a supervised model selection criterion to automatically identify a distance measure that can boost the performance of LP algorithm on a given dataset. This supervised criterion is the entropy of $c \times c$ distance matrix $D$ calculated on labeled data, where $D_{i,j}$ represents the average distance (cosine distance or JS distance) between examples from the $i$-th class and ones from the $j$-th class. It outperforms an unsupervised model selection criterion and a semi-supervised model selection criterion on SENSEVAL-3 data. Lower value of this supervised criterion implies larger inter-class distance and smaller intra-class distance in labeled data, which means clearer class structure. This will help LP algorithm to locate the correct class boundaries. But the unsupervised criterion cannot use the important label information, while the semi-supervised criterion may not properly predict the performance of learning algorithm if there is noise in labeled data. It may explain why this supervised criterion outperforms the other two model selection criteria.

## 6.3 Partially Supervised Sense Disambiguation by Learning Sense Number from Tagged and Untagged Corpora

In this work, we presented a partially supervised sense disambiguation algorithm, which performed sense disambiguation without the requirement that tagged corpus should include the instances of each possible sense of a target word.

Our algorithm tries to estimate the sense number of a target word in mixed data (tagged corpus+untagged corpus) by maximizing a stability criterion defined on classification result over all the possible values of sense number. At the same time, we can obtain a classification result on the mixed data. If the estimated sense number in the mixed data is equal to the sense number of the target word in tagged corpus, then there is no new sense in untagged corpus. Otherwise new senses will be represented by groups in which there is no instance from the tagged corpus. The stability criterion assesses the agreement between classification results on full mixed data and sampled mixed data. A partially supervised learning algorithm is used to classify mixed data into a given number of classes before stability evaluation.

Our ELP based partially supervised sense disambiguation algorithm was empirically compared with other related algorithms, e.g., SVM, a one-class partially supervised classification algorithm, and a clustering based partially supervised sense disambiguation algorithm. Experimental results on SENSEVAL-3 data indicated that the ELP based algorithm outperforms SVM and the partially supervised algorithm (LPU), and performs slightly better than semi-supervised k-means clustering based sense disambiguation algorithm. Furthermore, the ELP based algorithm performs more efficiently than the clustering based method.

## 6.4 Open Problems

In this thesis, we have investigated a series of novel learning algorithms to address the sense disambiguation problem, e.g. a GMM+MDL based clustering algorithm, a graph based semi-supervised classification algorithm, and a partially supervised classification algorithm.

But many issues are not attacked in this work, which may be left as open problems for the future work.

(1) Sense disambiguation is an intermediate task for natural language systems, e.g. machine translation, information retrieval, speech processing and text processing. This thesis evaluates WSD as a stand alone task. It will be more attractive if there are results on some end tasks relevant to actual applications.

(2) WSD is usually generalized as a classification task, which may be addressed by any off-the-shelf classification algorithm. Therefore, the success of corpus based sense disambiguation depends on the advances in machine learning research. There is still much more space to improve the learning algorithms in this thesis. The graph is the single most important quantity for graph-based semi-supervised learning. Parameterizing graph edge weights should be the first step of any graph-based semi-supervised learning methods. Current methods, e.g., evidence maximization, entropy minimization, are not efficient enough. Can we find better ways to learn the graph structure and parameters? Can we find better methods

to automatically select seeds for the ELP algorithm?

(3) There are a large amount of resources for sense disambiguation of English language or other western languages, e.g. WordNet, Semcor, SENSEVAL corpora, BNC, WSJ, etc. But the resources for other languages (e.g. Chinese language) are much less. There is some work in sense disambiguation [29, 74] that can make use of the raw corpora in the second language to help sense disambiguation in the first language. Inductive transfer or transfer learning has gained much attention in machine learning, which refers to the problem of retaining and applying the knowledge learned in one or more tasks to efficiently develop an effective hypothesis for a new task [131]. Can we use existing transfer learning methods or find better ways to transfer the learned knowledge from the language with rich resource to another language with poor resource?

We expect advances in research will address these questions. We hope that word sense disambiguation becomes a fruitful area for natural language processing.

# Bibliography

[1] Agirre, E., & Martinez, D. 2004. Unsupervised WSD Based on Automatically Retrieved Examples: The Importance of Bias. *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain.

[2] Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. 1998. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. *Proceedings of ACM SIGMOD Conference*(pp. 94–105), Seattle, Washington.

[3] Akaike, H. 1974. A New Look at the Statistical Model Identification. *IEEE Transactions on Automatic Control*, 19:716-723.

[4] Anderson, J.R. 1976. Language, Memory, and Thought. *Lawrence Erlbaum and Associates*, Hillsdale, New Jersey.

[5] Balcan,M. F., Blum, A., & Yang, K. 2005. Co-training and Expansion: Towards Bridging Theory and Practice. *Advances in Neural Information Processing Systems 17.*

[6] Basu, S., Banerjee, A., & Mooney, R. J. 2002. Semi-Supervised Clustering by Seeding. *Proceedings of 19th International Conference on Machine Learning.*

[7] Belkin, M., & Niyogi, P. 2002. Using Manifold Structure for Partially Labeled Classification. *Advances in Neural Information Processing Systems 15.*

[8] Ben-Hur, A., Elisseeff, A., & Guyon, I. 2002. A Stability Based Method for Discovering Structure in Clustered Data. *Pacific Symposium on Biocomputing*, pages 6-17.

[9] Bennett, K., & Demiriz, A. 1999. Semi-Supervised Support Vector Machines. *Advances in Neural Information Processing Systems 11.*

[10] Bie T.D., Momma M., Cristianini N. 2003. Efficiently Learning the Metric Using Side-Information. *Proceedings of the 14th International Conference on Algorithmic Learning Theory (ALT2003)*, Sapporo, Japan, Lecture Notes in Artificial Intelligence, Vol. 2842, pp. 175-189, Springer.

[11] Bilenko, M., Basu, S., & Mooney, R.J. 2004. Integrating Constraints and Metric Learning in Semi-Supervised Clustering. *Proceedings of the 21st International Conference on Machine Learning*, pp. 81-88, Banff, Canada.

[12] Black, E. 1988. An Experiment in Computational Discrimination of English Word Senses. *IBM Journal of Research and Development*, 32(2), pages 185-194.

[13] Blum, A., & Mitchell, T. 1998. Combining Labeled and Unlabeled Data with Co-training. *Proceedings of the Workshop on Computational Learning Theory.*

[14] Blum, A., & Chawla, S.. 2001. Learning from Labeled and Unlabeled Data Using Graph Mincuts. *Proceedings of the 18th International Conference on Machine Learning.*

[15] Blum, A., Lafferty, J., Rwebangira, R., & Reddy, R. 2004. Semi-Supervised Learning Using Randomized Mincuts. *Proceedings of the 21st International Conference on Machine Learning.*

[16] Bouman, C. A., Shapiro, M., Cook, G. W., Atkins, C. B., & Cheng, H. 1998. Cluster: An Unsupervised Algorithm for Modeling Gaussian Mixtures. *http://dynamo.ecn.purdue.edu/~bouman/software/cluster/.*

[17] Breckenridge, J. 1989. Replicating Cluster Analysis: Method, Consistency and Validity. *Multivariate Behavioural research*

[18] Brown P.F., Stephen, D.P., Vincent, D.P., & Mercer, R.L. 1991. Word Sense Disambiguation Using Statistical Methods. *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics.*

[19] Brown, P.F., Vincent D.P., deSouza, P.V., Lai, J.C., & Mercer, R.L. 1992. Class-based N-gram Models of Natural Language. *Computational Linguistics*, 18(4):467-479.

[20] Bruce, R., & Wiebe, J. 1994. Word Sense Disambiguation Using Decomposable Models. *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 139-145, Las Cruces, New Mexico, USA.

[21] Caraballo, A. S. 1999. Automatic Construction of A Hypernym-Labeled Noun Hierarchy from Text. *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics.*

[22] Castelli, V., & Cover, T. 1995. The Exponential Value of Labeled Samples. *Pattern Recognition Letters*, 16, 105-111.

[23] Castelli, V., & Cover, T. 1996. The Relative Value of Labeled and Unlabeled Samples in Pattern Recognition with an Unknown Mixing Parameter. *IEEE Transactions on Information Theory*, 42, 2101-2117.

[24] Chan, Y.S., & Ng, H.T. 2005. Word Sense Disambiguation with Distribution Estimation. *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, Edinburgh, Scotland, UK.

[25] Chapelle, O., Weston, J., & Schölkopf, B. 2002. Cluster Kernels for Semi-supervised Learning. *Advances in Neural Information Processing Systems 15.*

[26] Chen, J.Y. & Palmer, M. 2004. Chinese Verb Sense Discrimination Using an EM Clustering Model with Rich Linguistic Features. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics.*

[27] Collins, A. M., & Loftus, E. F. 1975. A Spreading Activation Theory of Semantic Processing. *Psychological Review*, 82(6), 407-428.

[28] Cozman, F., Cohen, I., & Cirelo, M. 2003. Semi-supervised Learning of Mixture Models. *Proceedings of the 20th International Conference on Machine Learning.*

[29] Dagan, I. & Itai A.. 1994. Word Sense Disambiguation Using A Second Language Monolingual Corpus. *Computational Linguistics*, Vol. 20(4), pp. 563-596.

[30] Dagan, I., Lee, L., & Pereira, F.. 1997. Similarity-Based Methods for Word Sense Disambiguation. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics.*

[31] Dash, M., & Liu, H. 1997. Feature Selection for Classification. *Intelligent Data Analysis*, Vol. 1, 131–156.

[32] Dash, M., & Liu, H.. 2000. Feature Selection for Clustering. *Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining*(pp. 110–121).

[33] Dash, M., Choi, K., Scheuermann, P., & Liu, H. 2002. Feature Selection for Clustering - A Filter Solution. *Proceedings of IEEE International Conference on Data Mining*, Maebashi City, Japan.

[34] Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, vol. 41(6):391-407, 1990.

[35] Dempster, A. P., Laird, N. M., & Rubin, D. B. 1977. Maximum Likelihood from Incomplete Data Using the EM Algorithm. *Journal of the Royal Statistical Society*, 39(B).

[36] Denis, F., Gilleron, R., & Tommasi, M.. 2002. Text Classification from Positive and Unlabeled Examples. *Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems.*

[37] Devaney, M., & Ram, A. 1997. Efficient Feature Selection in Conceptual Clustering. *Proceedings of the 14th International Conference on Machine Learning*(pp. 92–97), Morgan Kaufmann, San Francisco, CA.

[38] 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).

[39] Dorow, B, & Widdows, D. 2003. Discovering Corpus-Specific Word Senses. *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics, Conference Companion (research notes and demos)*(pp.79–82).

[40] Dy, J. G., & Brodley, C. E. 2000. Feature Subset Selection and Order Identification for Unsupervised Learning. *Proceedings of the 17th International Conference on Machine Learning*(pp. 247–254).

[41] Erk, K. 2006. Unknown Word Sense Detection as Outlier Detection. *Proceedings of NAACL 2006*, NYC, USA.

[42] Escudero, G., Marquez, L., & Rigau, G. 2000. An Empirical Study of the Domain Dependence of Supervised Word Sense Disambiguation Systems. *Proceedings of EMNLP/VLC00*, Hong Kong.

[43] Fisher, R.A. 1956. Statistical Methods and Scientific Inference. Olyver and Boyd.

[44] Forman, G. 2003. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research* 3(Mar):1289-1305.

[45] Fridlyand, J., & Dudoit, S. 2001. Applications of Resampling Methods to Estimate the Number of Clusters and to Improve the Accuracy of a Clustering Method. Technical Report 600, Statistics Department, UC Berkeley.

[46] Fukumoto, F., & Suzuki, Y. 1999. Word Sense Disambiguation in Untagged Text Based on Term Weight Learning. *Proceedings of the 9th Conference of European Chapter of the Association for Computational Linguistics*, pp. 209–216.

[47] Gale, W. A., Church, K. W., & Yarowsky, D. 1992. Using Bilingual Materials to Develop Word Sense Disambiguation Methods. *Proceedings of the International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 101-112.

[48] Gale, W. A., Church, K. W., & Yarowsky, D. 1993. A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities*, 26, 415-439.

[49] Gliozzo, A., Strapparava, C., & Dagan, I. 2004. Unsupervised and Supervised Exploitation of Semantic Domains in Lexical Disambiguation. Computer Speech and Language.

[50] Goldman, S., & Zhou, Y. 2000. Enhancing Supervised Learning with Unlabeled Data. *Proceedings of the 17th International Conference on Machine Learning.*

[51] 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.

[52] Hillel, A. B., Hertz, T., Shental, N., & Weinshall, D. 2003. Learning Distance Functions Using Equivalence Relations. *Proceedings of the 20th International Conference on Machine Learning.*

[53] Hindle, D. 1990. Noun Classification from Predicate-Argument Structures. *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics.*

[54] Hirst, G. 1987. Semantic Interpretation and the Resolution of Ambiguity. *Studies in Natural Language Processing*, Cambridge University Press, Cambridge, United Kingdom.

[55] Hirst, G., and St-Onge, D. 1998. Lexical Chains as Representations of Context in the Detection and Correction of Malaproprisms. *WordNet: An electronic lexical database*, MIT Press.

[56] Ide, N., & Véronis, J. 1998. Word Sense Disambiguation: The State of the Art. *Computational Linguistics*, 24:1, 1–41.

[57] Joachims, T. 1999. Transductive Inference for Text Classification Using Support Vector Machines. *Proceedings of the 16th International Conference on Machine Learning.*

[58] Joachims, T. 2002. Optimizing Search Engines using Clickthrough Data. *Proceedings of the ACM SIGKDD 2002.*

[59] Joachims, T. 2003. Transductive Learning via Spectral Graph Partitioning. *Proceedings of the 20th International Conference on Machine Learning.*

[60] Karov, Y., & Edelman, S. 1998. Similarity-Based Word Sense Disambiguation. *Computational Linguistics*, 24(1): 41-59.

[61] Kelly, E. F., & Stone, P. J. 1975. *Computer Recognition of English Word Senses*, North-Holland, Amsterdam.

[62] Kim, Y. S., Street, W. N., & Menczer, F. 2000. Feature Selection in Unsupervised Learning via Evolutionary Search. *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*(pp. 365–369).

[63] Krovetz, R., & Croft, W. B. 1992. Lexical Ambiguity and Information Retrieval. *ACM Transactions on Information Systems*, 10(2), 115-141.

[64] Klein, D., Kamvar, S. D., & Manning, C. 2002. From Instance-level Constraints to Space-Level Constraints: Making the Most of Prior Knowledge in Data Clustering. *Proceedings of the 19th International Conference on Machine Learning.*

[65] Lange, T., Braun, M., Roth, V., & Buhmann, J. M. 2002. Stability-Based Model Selection. *Advances in Neural Information Processing Systems 15.*

[66] Law, M. H., Figueiredo, M., & Jain, A. K. 2002. Feature Selection in Mixture-Based Clustering. *Advances in Neural Information Processing Systems 15.*

[67] Leacock, C., Miller, G.A. & Chodorow, M.. 1998. Using Corpus Statistics and WordNet Relations for Sense Identification. *Computational Linguistics*, 24:1, 147–165.

[68] Lee, W.S., & Liu, B.. 2003. Learning from Positive and Unlabeled Examples Using Weighted Logistic Regression. *Proceedings of the 20th International Conference on Machine Learning.*

[69] 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).

[70] Lehman, J. F. 1994. Toward the Essential Nature of Statistical Knowledge in Sense Resolution. *Proceedings of the 12th National Conference on Artificial Intelligence*, pages 734-741, Seattle, Washington, USA.

[71] Lesk, M. 1986. Automated Sense Disambiguation Using Machine-Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. *Proceedings of the 1986 SIGDOC Conference*, pages 24-26, Toronto, Canada.

[72] Leskes, B. 2005. The Value of Agreement, a New Boosting Algorithm. *Proceedings of the 18th Annual Conference on Computational Learning Theory.*

[73] Levine, E., & Domany, E. 2001. Resampling Method for Unsupervised Estimation of Cluster Validity. *Neural Computation*, Vol. 13, 2573–2593.

[74] Li, H. & Li, C.. 2004. Word Translation Disambiguation Using Bilingual Bootstrapping. *Computational Linguistics*, 30(1), 1-22.

[75] Li, X., & Liu, B. 2003. Learning to Classify Text Using Positive and Unlabeled Data. *Proceedings of the 18th International Joint Conference on Artificial Intelligence.*

[76] Lin, D.K. 1997. Using Syntactic Dependency as Local Context to Resolve Word Sense Ambiguity. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics.*

[77] Lin, D.K. 1998. Automatic Retrieval and Clustering of Similar Words. *Proceedings of COLING-ACL 98*, Montreal, Canada.

[78] Lin, J.H. 1991. Divergence Measures Based on the Shannon Entropy. *IEEE Transactions on Information Theory*, 37:1, 145–150.

[79] Liu, B., Lee, W.S., Yu, P.S., & Li, X. 2002. Partially Supervised Classification of Text Documents. *Proceedings of the 19th International Conference on Machine Learning.*

[80] Liu, B., Dai, Y., Li, X., Lee, W.S., & Yu, P.. 2003. Building Text Classifiers Using Positive and Unlabeled Examples. *Proceedings of the 3rd IEEE International Conference on Data Mining*, Melbourne, Florida.

[81] Manevitz, L.M., & Yousef, M. 2001. One Class SVMs for Document Classification. *Journal of Machine Learning*, 2, 139-154.

[82] Masterman, M. 1957. The Thesaurus in Syntax and Semantics. *Mechanical Translation*, 4, 1-2.

[83] Masterman, M. 1961. Semantic Message Detection for Machine Translation, Using an Interlingua. *International Conference on Machine Translation of Languages and Applied Language Analysis*, Her Majestyis Stationery Office, London.

[84] 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.*

[85] McClelland, J. L., & Rumelhart, D. E. 1981. An Interactive Activation of Context Effects in Letter Perception: Part 1. An Account of Basic Findings. *Psychological review*, 88, 375-407.

[86] Mihalcea, R., & Moldovan, D. 1999. An Automatic Method for Generating Sense Tagged Corpora. *Proceedings of the 16th National Conference on Artificial Intelligence and Eleventh Conference on Innovative Applications of Artificial Intelligence*, pages 461-466, Orlando, Florida, USA.

[87] Mihalcea R. 2004a. Co-training and Self-training for Word Sense Disambiguation. *Proceedings of the Conference on Natural Language Learning.*

[88] Mihalcea R., Chklovski, T., & Kilgariff, A.. 2004b. The Senseval-3 English Lexical Sample Task. *Proceedings of the 3rd International Workshop on the Evaluation of Systems for the Semantic Analysis of Text.*

[89] Mihalcea, R., Tarau, P., & Figa, E. 2004c. PageRank on Semantic Networks, with Application to Word Sense Disambiguation. *Proceedings of The 20th International Conference on Computational Linguistics*, Switzerland, Geneva.

[90] Mihalcea, R. 2005. Unsupervised Large-Vocabulary Word Sense Disambiguation with Graph-based Algorithms for Sequence Data Labeling. *Proceedings of the Joint Conference on Human Language Technology / Empirial Methods in Natural Language Processing*, Vancouver, Canada.

[91] Miller, G.A., Beckwith, R.T., Fellbaum, C.D., Gross, D., & Miller, K.J. 1990. WordNet: An On-line Lexical Database. *International Journal of Lexicography*, 3(4), 235-244.

[92] Mitchell, T. 1999. The Role of Unlabeled Data in Supervised Learning. *Proceedings of the Sixth International Colloquium on Cognitive Science.*

[93] Mitra, P., Murthy, A. C., & Pal, K. S. 2002. Unsupervised Feature Selection Using Feature Similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:4, 301–312.

[94] Modha, D. S., & Spangler, W. S. 2003. Feature Weighting in K-Means Clustering. *Machine Learning*, 52:3, 217–237.

[95] Mooney, R.J. 1996. Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning. *Proceedings of the 1996 Conference on Empirical Methods in Natural Language Processing*, pg. 82-91.

[96] Navarro, D.J., & Myung, I.J. 2004. Model Evaluation and Selection. B, Everitt & D. Howel (eds.), *Encyclopedia of Behavioral Statistics*. Wiley.

[97] Ng, A., Jordan, M., & Weiss, Y. 2001. On Spectral Clustering: Analysis and an Algorithm. *Advances in Neural Information Processing Systems 14.*

[98] Ng, H.T. & Lee, H.B.. 1996. Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-Based Approach. *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pp. 40-47.

[99] 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.

[100] Nigam, K., & Ghani, R. 2000. Analyzing the Effectiveness and Applicability of Co-training. *Proceedins of the Ninth International Conference on Information and Knowledge Management.*

[101] Nigam, K., McCallum, A. K., Thrun, S., & Mitchell, T. 2000. Text Classification from Labeled and Unlabeled Documents Using EM. *Machine Learning*, 39, 103-134.

[102] Nigam, K. 2001. Using Unlabeled Data to Improve Text Classification (Technical Report CMU-CS-01-126). Carnegie Mellon University. Doctoral Dissertation.

[103] Niu, Z.Y., Ji, D.H., & Tan, C.L. 2004. Document Clustering Based on Cluster Validation. *Proceedings of the 13th ACM International Conference on Information and Knowledge Management.*

[104] Niu, Z.Y., Ji, D.H., & Tan, C.L. 2005. Word Sense Disambiguation Using Label Propagation Based Semi-Supervised Learning. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics.*

[105] Niwa, Y., & Nitta, Y. 1994. Coocurrence Vectors from Corpora vs Distance Vectors from Dictionaries. *Proceedings of the 15th International Conference on Computational Linguistics*, pages 304-309, Kyoto, Japan.

[106] Pantel, P., & Lin, D. K. 2002. Discovering Word Senses from Text. *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*(pp. 613-619).

[107] 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.*

[108] 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.

[109] Pedersen, T. 2000. A Simple Approach to Building Ensembles of Naive Bayesian Classifiers for Word Sense Disambiguation. *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics.*

[110] Pereira, F., Tishby, N., & Lee, L. 1993. Distributional Clustering of English Words. *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics.*

[111] Pham, T. P., Ng, H. T., & Lee, W. S. 2005. Word Sense Disambiguation with Semi-Supervised Learning. *Proceedings of the 20th National Conference on Artificial Intelligence*, pages 1093-1098, Pittsburgh, Pennsylvania, USA.

[112] Phillips, W., & Riloff, E. 2002. Exploiting Strong Syntactic Heuristics and Co-Training to Learn Semantic Lexicons. *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing.*

[113] Pudil, P., Novovicova, J., & Kittler, J. 1994. Floating Search Methods in Feature Selection. *Pattern Recognigion Letters*, Vol. 15, 1119-1125.

[114] Rabinovich, A. 2005. Stability Based Model Order Selection in Clustering Problems. Technical Report, UCSD.

[115] Ratsaby, J., & Venkatesh, S. 1995. Learning from a Mixture of Labeled and Unlabeled Examples with Parametric Side Information. *Proceedings of the 8th Annual Conference on Computational Learning Theory.*

[116] Resnik, P. 1995. Disambiguating Noun Groupings with Respect to WordNet Senses. *Proceedings of the 3rd Workshop on Very Large Corpora*, Cambridge, Massachusetts, 54-68.

[117] Riloff, E. and Shepherd, J., 1999. A Corpus-Based Bootstrapping Algorithm for Semi-Automated Semantic Lexicon Construction. *Journal of Natural Language Engineering*, Vol. 5, No. 2, pp. 147-156.

[118] Riloff, E., Wiebe, J., & Wilson, T. 2003. Learning Subjective Nouns Using Extraction Pattern Bootstrapping.*Proceedings of the 7th Conference on Natural Language Learning.*

[119] Rissanen, J. 1978. Modeling by Shortest Data Description. *Automatica*, Vol. 14, 465–471.

[120] Rissanen, J. 1996. Fisher Information and Stochastic Complexity. *IEEE Transactions on Information Theory*, 42, 40-47.

[121] Rissanen, J. 2001. Strong Optimality of the Normalized ML Models as Universal Codes and Information in Data. *IEEE Transactions on Information Theory*, 47, 1712-1717.

[122] Roark, B. & Charniak, E.. 1998. Noun-phrase Co-occurrence Statistics for Semi-automatic Semantic Lexicon Construction. *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics.*

[123] Salton, G. 1968. *Automatic Information Organization and Retrieval*, McGraw-Hill, New York.

[124] Scholkopf, B. , Platt, J.C., Shawe-Taylor, J., Smola, A.J., & Williamson, R.C. 1999. Estimating the Support of a High-dimensional Distribution. Technical report, Microsoft Research, MSR-TR-99-87.

[125] Schütze, H., & Pedersen, J. 1995. Information Retrieval Based on Word Senses. *Proceedings of SDAIR'95*, Las Vegas, Nevada.

[126] Schütze, H. 1998. Automatic Word Sense Discrimination. *Computational Linguistics*, 24:1, 97–123.

[127] Schwarz, G. 1978. Estimating the Dimension of a Model. *Annals of Statistics*, 6:461-464.

[128] Seeger, M. 2001. Learning with Labeled and Unlabeled Data. *Technical Report*, University of Edinburgh.

[129] 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.

[130] Shi, J., & Malik, J.. 2000. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 888-905.

[131] Silver, D. L. 2005. NIPS Workshop on Inductive Transfer: 10 Years Later.

[132] Slonim, N., Friedman, N., & Tishby, N. 2002. Unsupervised Document Classification Using Sequential Information Maximization. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

[133] Sproat, R., Hirschberg, J., & Yarowsky, D. 1992. A Corpus-Based Synthesizer. *Proceedings of the International Conference on Spoken Language Processing*, Banff, Alberta, Canada.

[134] Stone, M. 1974. Cross-validatory Choice and Assessment of Statistical Predictions. *Journal of Royal Statistical Society*, 36, 111-147.

[135] Szummer, M., & Jaakkola, T. 2001. Partially Labeled Classification with Markov Random Walks. *Advances in Neural Information Processing Systems 14.*

[136] Talavera, L. 1999. Feature Selection as a Preprocessing Step for Hierarchical Clustering. *Proceedings of the 16th International Conference on Machine Learning*(pp. 389–397).

[137] Talavera, L. 2000. Dependency-Based Feature Selection for Clustering Symbolic Data. *Intelligent Data Analysis*, Vol. 4, 19-28.

[138] Tibshirani, R., Walther G., & Hastie, T. 2001a. Estimating the Number of Clusters via the Gap Statistic. *Journal of Royal Statistical Society B*, 63(2):411-423, 2001a.

[139] Tibshirani, R., Walther, G., Botstein, D., & Brown, P. 2001b. Cluster Validation by Prediction Strength. *Technical Report*, Statistics Department, Stanford University.

[140] Towel, G., & Voorheest, E.M. 1998. Disambiguating Highly Ambiguous Words. *Computational Linguistics*, 24:1, 125–145.

[141] Vaithyanathan, S., & Dom, B. 1999. Model Selection in Unsupervised Learning with Applications To Document Clustering. *Proc. of the 16th Int. Conf. on Machine Learning.*

[142] Vapnik, V. 1998. Statistical Learning Theory. Springer.

[143] Véronis, J, & Ide, N. 1990. Word Sense Disambiguation with Very Large Neural Networks Extracted from Machine Readable Dictionaries. *Proceedings of the 13th International Conference on Computational Linguistics*, vol. 2, pages 389-394, Helsinki, Finland.

[144] Véronis, J. 2004. HyperLex: Lexical Cartography for Information Retrieval. *Computer, Speech and Language*, 18:3, 223–252.

[145] Voorhes, E. M. 1993. Using WordNet to Disambiguate Word Senses for Text Retrieval. *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Pittsburgh, Pennsylvania, 171-180.

[146] Wagstaff, K., Cardie, C., Rogers, S., & Schroedl, S. 2001. Constrained K-Means Clustering with Background Knowledge. *Proceedings of the 18th International Conference on Machine Learning.*

[147] Weaver, W. 1949. Translation. *Locke, William N. and Booth, A. Donald (1955) (Eds.), Machine translation of languages.* John Wiley & Sons, New York, pp. 15-23.

[148] Weiss, S. 1973. Learning to Disambiguate. *Information Storage and Retrieval*, 9.

[149] Widdows, D. 2003. Unsupervised Methods for Developing Taxonomies by Combining Syntactic and Statistical Information. *Proceedings of the Human Language Technology / Conference of the North American Chapter of the Association for Computational Linguistics*(pp. 276–283).

[150] Wilks, Y. A. 1968. On-Line Semantic Analysis of English Texts. *Mechanical Translation*, 11(3-4), 59-72.

[151] Wilks, Y. A., Fass, D., Guo, C.-M., MacDonald, J. E., Plate, T., & Slator, B. A. 1990. Providing Machine Tractable Dictionary Tools. Pustejovsky, James (Ed.), *Semantics and the Lexicon*, MIT Press, Cambridge, Massachusetts.

[152] Wu, D., Su, W., & Carpuat, M. 2004. A Kernel PCA Method for Superior Word Sense Disambiguation. *Proceedins of the 42nd Annual Meeting of the Association for Computational Linguistics.*

[153] Xing, E., Ng, A. Y., Jordan, M., & Russell, S. 2003. Distance Metric Learning, with Application to Clustering with Side-Information. *Advances in Neural Information Processing System 16.*

[154] 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.

[155] 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.

[156] Yarowsky, D. 1997. Homograph Disambiguation in Text-to-Speech Synthesis. *Progress in Speech Synthesis*, Springer-Verlag, New York, 157-172.

[157] Yarowsky, D. 2000. Hierarchical Decision Lists for Word Sense Disambiguation. *Computers and the Humanities*, 34.

[158] Yarowsky, D., Cucerzan, S., Florian, R., Schafer, C., & Wicentowski, R. 2001. The Johns Hopkins SENSEVAL2 System Descriptions. *Proceedings of the 2nd International Workshop on Evaluating Word Sense Disambiguation Systems (SENSEVAL-2)*, pages 163C166.

[159] Yeung, D. S., & Wang, X. Z. 2002. Improving Performance of Similarity-Based Clustering by Feature Weight Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:4, 556–561.

[160] Yu, H., Han, J., & Chang, K. C.-C. 2002. PEBL: Positive Example Based Learning for Web Page Classification Using SVM. *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery in Databases.*

[161] 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.

[162] Zhou, D., Weston, J., Gretton, A., Bousquet, O., & Scholkopf, B. 2004. Ranking on Data Manifolds. *Advances in Neural Information Processing System 17.*

[163] Zhou, Z. H., & Li, M. 2005. Tri-training: Exploiting Unlabeled Data Using Three Classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17, 1529-1541.

[164] Zhu, X. & Ghahramani, Z.. 2002. Learning from Labeled and Unlabeled Data with Label Propagation. *CMU CALD tech report CMU-CALD-02-107.*

[165] 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.*

[166] Zhu, X. 2005. Semi-Supervised Learning with Graphs. *Ph.D. Thesis, also CMU LTI tech report CMU-LTI-05-192.*

# Appendix A

# List of Publications

Conference Papers:

Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan (2006). Partially Supervised Sense Disambiguation by Learning Sense Number from Tagged and Untagged Corpora. *Proceedings of EMNLP 2006*. Sydney, Australia.

Jinxiu Chen, Dong-Hong Ji, Chew Lim Tan, Zheng-Yu Niu (2006). Unsupervised Relation Disambiguation with Order Identification Capabilities. *Proceedings of EMNLP 2006*. Sydney, Australia.

Jinxiu Chen, Dong-Hong Ji, Chew Lim Tan, Zheng-Yu Niu (2006). Semi-supervised Relation Extraction With Label Propagation. *Proceedings of COLING/ACL 2006*. Sydney, Australia.

Jinxiu Chen, Dong-Hong Ji, Chew Lim Tan, Zheng-Yu Niu (2006). Unsupervised Relation Disambiguation With Model Order Identification. *Proceedings of COLING/ACL 2006*. Sydney, Australia.

Jinxiu Chen, Dong-Hong Ji, Chew Lim Tan, Zheng-Yu Niu (2006). Semi-supervised Relation Extraction With Label Propagation. *Proceedings of HLT/NAACL 2006*. New York, USA.

Yu Nie, Dong-Hong Ji, Lingpeng Yang, Zheng-Yu Niu, Tingting He (2006). Multi-document Summarization Using a Clustering Based Hybrid Strategy. *Proceedings of AIRS-2006*. Singapore.

Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan (2005). Word Sense Disambiguation Using Label Propagation Based Semi-supervised Learning. *Proceedings of ACL-2005*. Ann Arbor, USA.

Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan (2005). Semi-Supervised Feature Clustering with Application to Word Sense Disambiguation. *Proceedings of HLT/EMNLP 2005*. Vancouver, Canada.

Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan, Lingpeng Yang (2005). Word Sense Disambiguation by Local and Global Consistency Based Semi-supervised Learning. *Proceedings of CICLING-2005*. Mexico City, Mexico.

Jinxiu Chen, Dong-Hong Ji, Chew Lim Tan, Zheng-Yu Niu (2005). Automatic Relation Extraction with Model Order Selection and Discriminative Label Identification. *Proceedings*

*of IJCNLP-2005*. Jeju Island, Korea.

Jinxiu Chen, Dong-Hong Ji, Chew Lim Tan, Zheng-Yu Niu (2005). Unsupervised Feature Selection for Relation Extraction. *Proceedings of IJCNLP-2005*. Jeju Island, Korea.

Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan (2004). Document Clustering Based on Cluster Validation. *Proceedings of CIKM-2004*. Washington D.C., USA.

Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan (2004). Learning Word Senses With Feature Selection and Order Identification Capabilities. *Proceedings of ACL-2004*. Barcelona, Spain.

Zheng-Yu Niu, Dong-Hong Ji (2004). Feature Selection for Chinese Character Sense Discrimination. *Proceedings of CICLING-2004*. Seoul, Korea.

Journal Papers:

Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan (2007). Using Cluster Validation Criterion to Identify Optimal Feature Subset and Cluster Number for Document Clustering. *Information Processing and Management*, Volume 43, Pages: 730-739.

Lingpeng Yang, Dong-Hong Ji, Li Tang, Zheng-Yu Niu (2005). Chinese Information Retrieval Based on Terms and Relevant Terms. *ACM Transactions on Asian Language Information Processing*, Volume 4, Issue 3, Pages: 357-374.