A Study of Classifier Combination and Semi-Supervised Learning for Word Sense Disambiguation

by

LE ANH CUONG

submitted to
Japan Advanced Institute of Science and Technology
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Supervisor: Professor AKIRA SHIMAZU

School of Information Science
Japan Advanced Institute of Science and Technology

March 2007

To My Family

Abstract

Word Sense Disambiguation (WSD) involves the association of a polysemous word in a text or discourse with a particular sense among numerous potential senses of that word. This is an “intermediate task” necessary to accomplish most natural language processing tasks. It is obviously essential for language understanding application, such as message understanding and human-machine communication; it is also at least helpful for other applications whose aim is not language understanding, such as machine translation and information retrieval, among others.

The automatic disambiguation of word senses has been an interest and concern since the 1950s [Ide *et al.* (1998)]. Although there have been many studies investigated on various methods for this problem, the performance of available WSD systems or published results are limited (accuracy around 70%). Therefore, WSD is still an open problem and is a challenge in Natural Language Processing (NLP) community. Nowadays, with the strong and fast development of machine learning methods and their success in applying to many NLP tasks, the use of machine learning techniques in WSD has been becoming more interest and attractive. This thesis also lies in this research direction, in which we present a study of classifier combination and semi-supervised learning for WSD. In addition, we also work on context representation and feature selection which play important roles in obtaining high accuracy of WSD task. Particularly, the following three problems are targeted in this research.

- The first problem is that of determining useful information for detecting word senses. Concerning this problem, there is a fact that if a classifier contains more useful information, as well as we can extract more useful information from test patterns, then the classifier can recognize patterns more correctly. Therefore, determining useful information for both training and test data plays an important role in a classification. Regarding to WSD, we consider two aspects including context presentation and feature selection. For the former, we will investigate various kinds of knowledge and present them as feature subsets. For the latter, feature selection methods are used to obtain the best combination of these subsets, and to select useful individual features. At the end, we obtain a set of selected features in order to get supervised WSD systems with high accuracy.

- The second problem is to improve performance of supervised WSD systems by using classifier combination techniques. As well known, a classification algorithm used with a specific set of features may not be appropriate with a different set of features. In addition, classification algorithms are different in their theories, and hence achieve different degrees of success for different applications. As different classifiers may offer complementary information about the patterns to be classified, combining classifiers in an efficient way, therefore, can achieve better classification results than any single classifier (even the best one). In this work, numerous various combination rules will be investigated and experimented. In particular, beside applying several common rules such as majority voting and average rule, we will develop new frameworks of classifier combination for WSD based on Dempster-Shafer theory of evidence and Ordered Weighted Aggregating (OWA, for short) operators. Experiments have shown that combining classifiers significantly improves performance of supervised WSD systems.

- The third problem is to boost supervised WSD by exploiting unlabeled data. The process of using both labeled and unlabeled data to build a classifier is called semi-supervised learning. This work is motivated by an observation that labeled data is expensive and time consuming to build, while unlabeled data is cheap and easy to collect. In this thesis, we follow the semi-supervised learning approach, in which labeled data is iteratively extended from unlabeled data. We first explicitly identify inherent problems in this approach, and then propose corresponding solutions for them. This results in several new variants of the general bootstrapping algorithm. The experimental results show that the proposed solutions improve the conventional bootstrapping algorithms (particulary for self-training and co-training), and at the same time it is shown that unlabeled data is effective in increasing accuracy of supervised WSD systems.

In summary, the work in this thesis has concentrated on the two important tasks which have strong impacts on improving accuracy for WSD systems, including knowledge determination and using machine learning approaches. We have analyzed, investigated, and then provided solutions for the selected problems occurring in these tasks. The contributions of this thesis are expressed in both aspects, the theoretical study in developing machine learning methods and the empirical study in improving accuracy of WSD systems. Our experiments were conducted on standard datasets and their outputs were compared with those of state-of-the-art WSD systems in such a way that the research is kept competitive and most up-to-date.

Key words: Computational Linguistic, Word Sense Disambiguation, Feature Selection, Classifier Combination, Semi-supervised Learning.

Acknowledgments

First of all, I wish to express my respect and my deepest thanks to my advisor Professor Akira Shimazu, School of Information Sciences, JAIST for his kindly guidance, warm encouragement, and helpful supports before and during this research. He has given me much invaluable knowledge not only how to formulate a research idea or to write a good paper but also the vision and much useful experience in the academic life. I gratefully appreciate his patient supervision, and I am really lucky and proud to be one of his students.

I wish to say grateful thanks to Associate Professor Kiyoaki Shirai for his useful comments in some discussions. My study field is very closed to his research, so he gives me more confidence in my work.

I wish to say sincere thanks to Professor Yoshiteru Nakamori for his help in my subtheme research. He has given me as good as possible conditions for my work during this time.

I would also like to say my special thanks to Professor Ho Tu Bao for his valuable discussions and his supports. I also want to express my gratitude to him for all his helps not only for my study but also for my life from my first day to now in JAIST.

I wish to say sincere thanks to Professor Yuji Matsumoto and Professor Satoshi Tojo for serving as members of my dissertation committee and give me valuable comments. Professor Yuji Matsumoto has also given me valuable discussions and helped me much during the time I was an exchange student in NAIST in the year 2001.

I specially wish to say grateful thanks to Dr. Huynh Van Nam for all his helps during this research. His valuable comments and advices on research problems as well as the technical insights really help me much. Without his help my work can not run well.

I specially wish to express my thanks to Dr. Dam Hieu Chi for all his helps for my life and my study. I have had deeply discussions with him about research problems as well as others.

I would also like to say my special thanks to my friend, Dr. Nguyen Le Minh, for his valuable discussions and his help in my research.

I would also like to express my grateful appreciation to my former advisors, Professor Ha Quang Thuy and Professor Dinh Manh Tuong, College of Technology, Vietnam National University, Hanoi (VNUH), for their kindly recommendations and constant encouragement before and during my research at JAIST. Without their helps, I could not receive the permission to go to JAIST.

I also appreciate the help and the encouragement from Professor Ho Si Dam, Dr. Hoang Xuan Huan, Dr. Pham Hong Thai, and many other faculty members of College of Technology, VNUH.

I have received a lot of help from colleagues and friends in Shimazu-Lab and Shirai-Lab during last three years. Let me say special thanks to Dr. Makoto Nakamura, and Mr. Kenji Takano for all their helps for my life from the first day I came to Japan. I

also would like to thank my colleagues in Shimazu-Lab, Mr. Nguyen Phuong Thai, Mr. Nguyen Van Vinh, Mr. Nguyen Tri Thanh, and others for sharing their research ideas, useful experience, as well as valuable discussions and comments.

I also wish to send my deep acknowledgements to the GRP program for supporting me during the past three years; the International Information Science Foundation for supporting me to attend international conferences; JAIST staff for their kind and convenient procedures and services. I am really indebted to them.

I specially express my thanks and my respect to my former advisor, Dr. Pham Hong Nguyen. He has guided me to the way of research, specially in Natural Language Processing. I am very proud to be one of his students, and also a member in his group that researches about Machine Translation.

Last, but not least, my family is really the biggest motivation behind me. My wife Cao Thi Minh Nghia, my son Le Cao Anh Minh, my daughter Le Bao Anh, my dear parents, my dear brother's family, together with their unconditional sacrifices, love, and supports are always endless sources of inspiration for me to move forwards.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter we briefly state the research context, our motivations, as well as the major contributions of this thesis. Firstly, we briefly introduce the problem of word sense disambiguation and its important role in natural language processing. Secondly, we state the research problems which this thesis attempts to solve as well as the main motivations behind the work. Next, the main contributions of the thesis are shortly mentioned. Finally, the structure of the thesis will be outlined.

1.1 An Overview

Natural language processing (NLP) involves resolution of various type of ambiguity. Lexical ambiguity is one of these ambiguity types, and occurs when a single word (lexical form) is associated with multiple senses or meanings. The task of a word sense disambiguation system is to resolve the lexical ambiguity of a word in a given context.

1.1.1 Word Sense Disambiguation (WSD)

In general, there are two types of lexical ambiguity, that include morpho-syntactic ambiguity and sense ambiguity. Since the earliest day of WSD work there has been general agreement that these problems can be discriminative and become dependent problems in current researches. Therefore, word sense disambiguation has since focused largely on distinguishing sense among homographs belonging to the same syntactic category.

Polysemy and Homonymy

Within lexical ambiguity, one can distinguish between polysemy and homonymy. Homonyms are distinct units in the lexicon with identical phonetic ("homophones") and/or orthographic ("homographs") shape. Each of the homonyms has its own semantics. An example is *bank* ("financial institution") and *bank* ("slope"). In this example, homophony and homography cooccur. But this need not always be the case, compare *peak* [pi:k] ("summit") and *peek* [pi:k] ("glance"), which are homophones but not homographs. According to the diachronic criterion for homonymy, homonyms cannot be traced back to a common etymological origin. Compare for instance bat: the word for the animal has a Swedish origin, the word for the instrument is related to battle. According to the synchronic criterion, homonyms have no common semantics.

If multiplicity of meanings occurs distributed over several lexical items with the same shape, one speaks of homonymy; if it occurs within a single lexical item, it is polysemy. The different meanings of a polysemous expression have a base meaning in common. Furthermore, the meanings of a polysemous term are often related by means of metaphor or metynomy. An example is *point*: “punctuation mark”, “sharp end”, “detail, argument” etc. Here we observe several meanings within one lexical unit, the common base meaning of which could be something like “smallest unit”. Either this unit is concrete (“punctuation mark”, “sharp end”) or it is a metaphorical unit as in “detail, argument”. There are cases of idiosyncratic polysemy, e.g. with *green* (“a certain color”, and “inexperienced”) or with *point* (see above). But there are also cases of systematic polysemy, e.g. the actual/ dispositional distinction (e.g. with *fast* in *this is a fast car*) or the building/ institution distinction (e.g. with *school*).

WSD is the task of disambiguating polysemy, in which **polysemy** characterizes words that have more than one meaning. It means that a single lexical item (same orthographic) has multiple meanings (senses). A such lexical item is called a **polysemous** word. In this consideration, the phenomenon of homonymy in which homonyms are in the same orthographic (“homographs”) shape is also a case of polysemy. Polysemy is an intrinsic property of words (in isolation from text), whereas “ambiguity” is a property of text. Whenever there is uncertainty as to the meaning that a speaker or writer intends, there is ambiguity. So polysemy indicates only potential ambiguity, and context works to remove ambiguity.

For a polysemous word, there is not the unique definition of its senses through different sources (e.g. different dictionaries). Different definitions can be based on different objectives or different sense-makers. Furthermore, word senses is in principle infinitely variable and context sensitive. It does not divide up easily into distinct sub-meanings or senses. Lexicographers frequently discover in corpus data loose and overlapping word meanings, and standard or conventional meaning extended, modulated, and exploited in a bewildering variety of ways.

WSD Task

This thesis follows a corpus-based approach, in which for a polysemous word we are given a set of sense-tagged examples for training and the task is to detect senses for sense-untagged examples of this polysemous word. The corpora we deal with in this thesis are English Lexical Samples of Senseval-2 and Senseval-3 [1]. There are 73 lexical items and 57 lexical items in English lexical samples of Senseval-2 and Senseval-3, respectively. In these datasets, according to the task description of Senseval organization, senses are organized in sense hierarchy or sense grouping to allow for fine-grained or coarse-grained sense distinctions to be used in scoring. At a coarse-grain a word often has a small number of senses that are clearly different and probably completely unrelated to each other, usually called *homographs*, for example *bank* in *Bank of England* and *bank* in *river bank* (it is a kind of homonymy). Such senses are just “accidentally” collected under the same word string. Regarding the objective of finer-grained distinctions the coarse-grained senses break up into a complex structure of interrelated senses, involving phenomena such as general polysemy, regular polysemy, and metaphorical extension. Thus, most sense distinctions are not as clear as the distinction between *bank* as 'financial institution'

[1] see http://www.senseval.org/

Table 1.1: Senses of the word "bank"

Sense ID	Definition & Examples
bank%1:04:00::	a flight maneuver; aircraft tips laterally about its longitudinal axis (especially in turning); "the plane went into a steep bank"
bank%1:06:00::	a building in which commercial banking is transacted; "the bank is on the corner of Nassau and Witherspoon"
bank%1:06:01::	a container (usually with a slot in the top) for keeping money at home; "the coin bank was empty"
bank%1:14:00::	a financial institution that accepts deposits and channels the money into lending activities; "he cashed a check at the bank"
bank%1:14:01::	an arrangement of similar objects in a row or in tiers; "he operated a bank of switches"
bank%1:17:00::	a long ridge or pile; "a huge bank of earth"
bank%1:17:01::	sloping land (especially the slope beside a body of water); "they pulled the canoe up on the bank"
bank%1:17:02::	a slope in the turn of a road or track; the outside is higher than the inside in order to reduce the effects of centrifugal force
bank%1:17:02::	a slope in the turn of a road or track; the outside is higher than the inside in order to reduce the effects of centrifugal force
bank%1:21:00::	a supply or stock held in reserve for future use (especially in emergencies)
bank%1:21:01::	the funds held by a gambling house or the dealer in some gambling games; "he tried to break the bank at Monte Carlo"

and *bank* as 'river side'. For example, *bank* as financial institution can split into the following cloud of related senses: the company or institution, the building itself, the counter where money is exchanged, a fund or reserve of money, a money box, the funds in a gambling house, the dealer in a gambling house, and a supply of something held in reserve (according to WordNet 2.1).

Table 1.1 shows sense definitions of the polysemous word *bank* in Senseval-3. Note that, in this example, the senses are distinguished at fine-grain, and if we want to focus on disambiguating these senses at coarse-grain then we map every sense to its parents (in this case, sense "bank%1:14:00::" is a hyponym of sense "bank%1:06:00::", and sense "bank%1:21:01::" is a hyponym of "bank%1:06:01::"). In this thesis, like other studies using these corpora we will evaluate our method with fine-grained scoring.

1.1.2 Applications of WSD

For application which are sensitive to semantic denotation, or more precisely lexical semantics, lexical ambiguity can pose a major obstacle. Resolution of lexical ambiguity, which is commonly termed "word sense disambiguation" (WSD), is expected to improve the quality of the such following research fields.

Machine Translation (MT). WSD is required for lexical choice in MT for words that have different translations for different senses and that are potentially ambiguous within a given domain (since non-domain senses could be removed during lexicon development).

For example, in an English-French financial news translator, the English noun *change* could translate to either *changement* ('transformation') or *monnaie* ('pocket money'). In MT, the senses are often represented directly as words in the target language.

Information Retrieval (IR). Ambiguity has to be resolved in some queries. For instance, given the query "*depression*" should the system return documents about illness, weather systems, or economics? A similar problem arises for proper nouns such as *Raleigh* (bicycle, person, city, etc.). Current IR systems do not use explicit WSD, and rely on the user typing enough context in the query to only retrieve documents relevant to the intended sense (e.g. "*tropical depression*"). Early experiments suggested that reliable IR would require at least 90% disambiguation accuracy for explicit WSD to be of benefit [Sanderson (1994)]. More recently, WSD has been shown to improve cross-lingual IR and document classification [Vossen *et al.* (2006)]. Besides document classification and cross-lingual IR, related applications include news recommendation and alerting, topic tracking, and automatic advertisement placement.

Information Extraction (IE) and Text Mining. WSD is required for the accurate analysis of text in many applications. For instance, an intelligence gathering system might require the flagging of, say, all the references to illegal *drugs*, rather than medical *drugs*. Bio-informatics research requires the relationship between genes and gene products to be catalogued from the vast scientific literature; however, genes and their proteins often have the same name. More generally, the Semantic Web requires automatic annotation of documents according to a reference ontology: all textual references must be resolved to the right concepts and event structures in the ontology: all textual references must be resolved to the right concepts and event structures in the ontology [Dill *et al.* (2003)]. Name-entity classification, co-reference determination, and acronym expansion (MG as *magnesium* or *milligram*) can also be cast as WSD problems for proper names. WSD is only beginning to be applied in these areas.

Lexicography. Modern lexicography is corpus-based, thus WSD and lexicography can work in a loop, with WSD providing rough empirical sense groupings and statistically significant contextual indicators of sense to lexicographers, who provide better sense inventories and sense-annotated corpora to WSD. Furthermore, intelligent dictionaries and thesauri might one day provide us with a semantically-cross-referenced dictionary as well as better contextual look-up facilities.

1.1.3 Corpus-Based WSD and Task Description

WSD is essentially a classification problem: given a polysemous word, its possible senses are considered as classes, and each occurrence of the word should be assigned to one or more of its possible classes based on the information extracted from the context where the word appears. This is the traditional and common characterization of WSD that considers it as an explicit process of disambiguation with respect to a fixed inventory of word senses.

Note that while most systems interpret only one polysemous word in the input (called the single-word task), other systems simultaneously interpret all polysemous words appearing in the input (called the all-words task). As the all-word task is just a set of independent single-word tasks, so in this work we will deal with the single-word task.

We follow the corpus-based (or data-driven) approach, in which a set of sense-tagged examples of a polysemous word is given and treated as the training data. The task is then to build a classifier based on the training data to detect senses for untagged examples of this polysemous word. Figure 1.1 shows the general scheme of corpus-based methods for WSD.

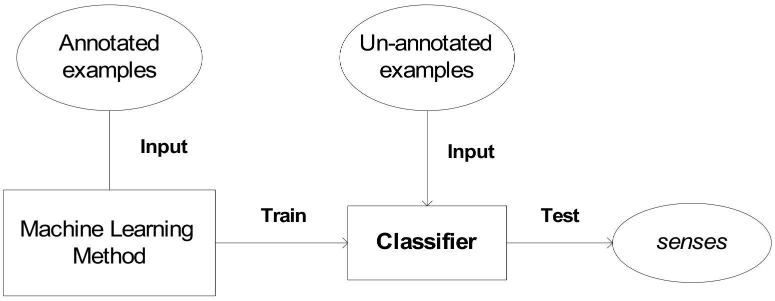


Figure 1.1: A General Scheme of Corpus-Based Methods for WSD

For example, suppose that we are dealing with the polysemous word *bank* in its part-of-speech *noun*, we denote this word by *bank.n* (this example is extracted from Senseval-3 data). We are given a set of examples (contexts) of *bank.n*, such that in each given example the word *bank.n* is assigned with its right sense. The following are two among hundreds of training contexts:

senseid="**bank%1:17:01::**"

<context>

All the grass grinds their little molars flat. So, something nest building. Possibly aligned to water a sort of <head>**bank**</head> by a rushing river. Perhaps something as natty and camp as Ratty in The Wind in the Willows. Dapper, dear, Noel Coward.

</context>

senseid="**bank%1:14:01::**"

<context>

It was late March. The air was raw and threatened rain but was tinged with the warmth of spring. The sky was a murky, pinkish grey; clouds swirled across it exposing higher, greyer <head>**banks**</head> of cloud. She snipped crisp green stalks with a pair of scissors. Milky liquid oozed from the stalks.

</context>

From the training data, we can build the sense inventory of *bank.n*, as shown in Table 1.1. One important point in the corpus-based approach is that the inventory of word senses is determined as the whole senses appearing in the training data, which is different from knowledge-based approach where the inventory of word senses is determined from a dictionary. Now, the task is to determine the right sense of *bank.n* (among its potential senses) for new contexts in test data, for example, for the following context:

<context>

> Original gravity can be roughly translated into alcoholic strength as follows: a 1036 beer has approximately 3.6 percent alcohol, a 1050 beer has 5 percent alcohol and so on. How to ruin perfection: gas connected to a cask of beer keeps the ale under a blanket of CO2, making it unpleasantly fizzy Serving the perfect pint: a fine <head>**bank**</head> of hand-pumps in a traditional tap room. Examples of different types of electric pumps. The two on the left are metered pumps serving exact half - pints, the ones on the right are free flow pumps.
>
> </context>

In this test example, we try to find out the right sense of *bank.n*, that is " bank%1:14:01::". Note that, the right senses of test examples are provided in a key-file for evaluation (scoring).

The major objective of this thesis is to applying and developing machine methods to improve accuracy of WSD. To this end, we focus on two machine learning aspects: classifier combination and semi-supervised learning, which aim to boost supervised WSD systems (the systems which use supervised learning algorithms such Naive Bayes, Maximum Entropy Models, Support Vector Machines, etc. to build WSD classifiers). In addition, we also work on context representation and feature selection which play important roles in obtaining high accuracy of WSD. The motivations and proposed solutions for the problems mentioned in this thesis are based on investigating and solving the limitations of related WSD studies. Experimental results were conducted on standard datasets (Senseval-2 and Senseval-3) and were compared to state-of-the-art systems in the field.

1.2 Motivations and Problems

1.2.1 Context Representation and Feature Selection

In the corpus-based approach, most studies just consider the information extracted from the context in which the target word appears. From our observation and other investigation such as [Klein *et al.* (2002)], we see that designing features plays an important role for classifiers to obtain high accuracy. Further, we observed that, in the context of WSD, the choice of what method of feature selection to use may more strongly influence the quality of classifiers than the choice of what machine learning algorithm to apply.

In previous studies, [Ng & Lee (1996), Lee & Ng(2002)] determined the feature set by listing features of some kinds of knowledge (called knowledge sources), while others just re-used their selections or with some modifications (add and/or remove more knowledge sources). In our opinion, this selection seems to be heuristic, and lack of a reasonable explanation supporting for this selection. This observation motivates us to study on applying learning approaches to the selection of knowledge sources as well as the selection of individual features. This work can be considered as the work of context representation and feature selection. In particular, we will focus on the following issues.

Selection of Knowledge Sources There are various knowledge sources appeared in the context of a polysemous word which can be considered as evidences for disambiguating word senses, such as morphological forms of surrounding words, syntactic relationships, or position relationship. We first collect as many as possible different knowledge sources, which we think useful for determining word senses. Naturally, the question arises here is that, should we use all these kinds of information or just some of them for the task? Therefore, we aim at finding out such a way that can explain why a selection is more appropriate than others.

Individual Feature Selection Feature selection is motivated by an observation that in some pattern recognition problems, there are too many features that require the cost for computation time and storage memory. In addition, some features may be redundant or noise. Therefore, the work on removing some features to reduce the size of the feature set while retaining the accuracy of classifiers is important and significant. This situation may also appear in WSD problem, so it motivates us to address the question whether applying feature selection methods for WSD is effective? There is a few of previous WSD studies regarding this issue, such as [Lee & Ng(2002), Mihalcea (2004), Pham *et al.* (2005)]. In these studies, the features which have frequencies equal or greater than 3 were selected. However, in this thesis, we will show that this selection much decreases the accuracy of WSD classification. We will also propose the use of a filter method in feature selection approaches to this task, making use of two well-known information measures of features including frequency and information-gain.

1.2.2 Applying and Developing Learning Methods

Machine learning is always an essential factor in corpus-based approaches. Since corpus-based WSD have been studied for more than one decade, many aspects of machine learning have been investigated and applied. Among them, various supervised learning algorithms have been used, such as in [Mooney (1996), Lee & Ng(2002), Ngai *et al.* (2004)]. For other aspects in machine learning, classifier combination and semi-supervised learning currently have attracted many researchers in WSD community. Application of these approaches has improved accuracy of supervised WSD, for example, see [Klein *et al.* (2002)] for advantages of classifier combination, and see [Pham *et al.* (2005)] for advantages of semi-supervised learning. Even thought, there are some limitations in these studies, which motivated us to do a deeper study on these approaches, as pointed out below.

Classifier Combination As observed in studies of machine learning systems, though one of the available learning systems could be chosen to achieve the best performance for a given pattern recognition problem, the set of patterns misclassified by the different classification systems would not necessarily overlap. This means that different classifiers may potentially offer complementary information about patterns to be classified. This observation highly motivated the recent interest in combining classifiers. Some studies of using classifier combination for WSD, such as [Klein *et al.* (2002)], have applied some combination strategies to obtain better results in comparison with individual supervised systems. However, these studies used only several combination strategies, such as majority voting, weighted voting, and simple mixture models. Some others combination strategies such as Product, Max, Min rules [Kittler *et al.* (1998)] have not been investigated. Furthermore, approaches in previous studies seem to be lack of describing a general combination framework accompanying with a corresponding theoretical basis, which should become the fundamental for applying the combination rules used. These observations motivate us to apply multiple classifier combination to WSD with a comprehensive collection of combination rules. Moreover, these rules would be used in combination frameworks corresponding to certain theoretical basis.

In order to generate individual classifiers, some studies used different supervised learning algorithms, such as [Klein *et al.* (2002)], and others used different feature spaces such as in [Wang & Matsumoto (2004), Pedersen (2000)]. In this thesis, we will investigate these both types of individual classifiers, in which we will use the different kinds of knowledge (information) to generate individual classifiers of the second type.

In addition, we will present a new scheme for classifier combination, in which one more phase of combination is added to the end of common combination strategies. This is motivated by the observation that different results yielded by applying various combination rules on the set of individual classifiers can be again used for one more combination. This new phase of combination is called second-layer combination, and thus the previous combination can be considered as the first-layer combination. This proposal is also motivated by the observation that combination usually gives better result in comparison with individual classifiers, so the second-layer combination is hoped to yield better results in comparison with the first-layer combination. Note that, this scheme can be considered as the generalization of some special cases in previous studies, such as [Klein *et al.* (2002), Florian & Yarowsky (2002), Wang & Matsumoto (2004)], among them [Klein *et al.* (2002)] used maximum entropy models, [Florian & Yarowsky (2002)] used median/average rule and majority voting, and [Wang & Matsumoto (2004)] used kNN, for the second-layer combination.

Semi-Supervised Learning As well known, labeled data takes long time and is expensive to obtain while unlabeled data is cheap and easy to collect. This observation motivates the use of unlabeled data to enhance the performance of the supervised classifier built on the initial labeled data. However, there are only several studies working on exploiting unlabeled data for WSD, such as [Ando(2006), Pham *et al.* (2005), Mihalcea (2004)]. In this work, we follow the approach, in which labeled data is iteratively extended from unlabeled data. This approach is encouraged by a natural observation that: if we have some labeled examples at the beginning, which are considered as seed examples, we can built a classifier and use it to assign labels for unlabeled examples. Performing this process iteratively we can obtain an extended labeled data which is hoped to strengthen the initial supervised classifier. In this approach, we explicitly identify and focus on the three following problems:

- The first problem regards the imbalance of labeled (training) data. We observe that if a classifier is built based on training data with a bias on certain classes (i.e., one or several classes dominate others), then this bias may become stronger at each extension of the labeled dataset. This is because a classifier tends to detect examples of dominant classes with high confidence, and consequently these examples are prioritized for a new set of labeled examples. Through steps of extending labeled data, the imbalance of labeled data may be increased, which may result in decreasing the accuracy of the initial classifier. Previous studies just solved this by fixing the number of new labeled examples for each class, such as in [Blum & Mitchell (1998), Pierce & Cardie (2001)]. However, this can not be implemented in some certain circumstances, for example in the case when we can not achieve enough confident new labeled examples of a class for the corresponding number which is pre-defined.

- The second problem is that of how to determine a subset of new labeled examples with high confidence. It is clear that adding a large number of misclassified examples into the labeled dataset will probably result in generating a poor classifier in the end. Therefore, one aims at obtaining new labeled examples with the highest accuracy possible. To reach this target, previous studies normally used the so-called threshold-based selection of new labeled examples. In particular, given a new example which is assigned a label with a probability of detection, a threshold value for this probability is predefined to decide whether a new labeled example will be selected or not, such as in [Yarowsky (1995), Blum & Mitchell (1998), Collins & Singer (1999)]. However, this threshold-based method of selection may lead to a situation where choosing a higher threshold will create difficulty in extending labeled data, while it does not always result in correct classification. By contrast, a lower threshold may result in more misclassified examples, but allows more new labeled examples to be added. Therefore, the determination of a "correct" threshold in the approach becomes an important issue. In addition, determining a commonly used threshold for all unknown data is also inappropriate.

- The third problem is that of how to generate the final classifier when the process of extending labeled data is completed. This process will be stopped when the number of iterations reaches a pre-specified value, or when the unlabeled dataset becomes empty. Normally, the classifier built on the labeled data obtained at the last iteration is chosen as the final one. Some studies use a development dataset to find the most appropriate value for the number of iterations, such as in [Pham *et al.* (2005), Mihalcea (2004)]. As mentioned in second problem, the last classifier may be built based on new training data with some misclassified examples, so both advantages and disadvantages are concurrently brought to the last classifier with respect to the initial classifier (built on the original dataset). Thus, choosing the classifier trained on the last labeled dataset as the final classifier may not always be a good solution. This observation suggests that we should combine the initial classifier and the last classifier to utilize advantages of both of them.

By reviewing various related studies, especially regarding the WSD problem, we found that most previous studies did not pay adequate attention to these three problems. In this work, we consider simultaneously these three problems for the objective of improving semi-supervised learning.

1.3 Main Contributions

As stated earlier, this thesis focuses on: (1) context representation and feature selection; (2) applying classifier combination techniques; and (3) exploiting unlabeled data. The main contributions of this thesis are summarized as follows:

1.3.1 Context Representation and Feature Selection

The uses of various kinds of knowledge (information) with the corresponding representations have been investigated. A method based on Forward and Backward Sequential Selection algorithms was proposed to select the best combination of these knowledge sources (i.e. context representations or feature subsets). Note that, different to previous studies, we simultaneously used three representations of bag-of-words corresponding to three window sizes including small, medium, and large. In addition, we proposed the use of a filter-based method to select useful individual features. In order to evaluate features, we used two measures including the frequency of features, and the information-gain of each feature obtained from training data. The experiment has shown that this feature selection gives better results in comparison with feature selections from previous studies.

At the end, the proposed approach has given us a certain confidence level on the selected features that is still lack in previous studies. Some results of this work are reported in [Le & Shimazu (2004)].

1.3.2 Classifier Combination

In this thesis, we have presented a general framework for combining classifiers for WSD in which individual classifiers use different representations of context or different learning algorithms. Several common rules used in previous studies, such as majority voting, weighted voting, and average rule are formulated in this framework.

On the one hand, various ways of using the context or different learning algorithms could be considered as providing different information sources to identify the meaning of the target word. Moreover, each of these information sources does not by itself provide 100% certainty as a whole piece of evidence for identifying the sense of the target. Under such an observation, we have interpreted the framework of classifier combination in terms of Dempster-Shafer theory of evidence [Shafer (1976)], and then formulated a general rule of classifier combination from which several interesting classifier combination schemes are derived. Particularly, applying the Dempster rule of combination (it is also considered as the orthogonal sum) on this framework, we achieve a new combination rule called DS combination rule. Furthermore, we also applied the discount operator on this framework to derive Discounting-and-Orthogonal sum rule. Under this combination strategy, we also can yield other combination rules, such as Average rule and Discounting-and-Average rule.

On the other hand, by considering individual classifiers as experts who have their own soft decisions (for instance, probability distributions over the set of classes) on the word sense identification, we now face with the problem how to derive a consensus decision based on their individual decisions. Intuitively, in such a situation, one (i.e., decision maker) may have a decision making strategy based on linguistic quantifiers, for instance "a decision should be finally selected if LQ experts have supported it", where LQ is a linguistic quantifier such as *all*, *most*, *at least half*,...". We have mimicked this decision making behavior of human beings for decision fusion in the context of WSD. In particular, we have used OWA operators for classifier fusion in their semantic relation to linguistic quantifiers [Zadeh 1983]. Under such a formulation, we provided a framework for combining classifiers, which also yields several commonly used decision rules for WSD. In particular, some fuzzy majority voting have been derived, which correspond to the combination rules, such as Max rule, Min rule, and Median rule. The use of fuzzy linguistic quantifiers not only help deriving but also provides a human-like interpretations to these rules. Note that, some of these combination rules are also yielded in the work of [Kittler *et al.* (1998)], but with some strong assumptions which is difficult to be accepted in WSD problem. Moreover, this approach does provide us a clear interpretation about the semantics of these combination rules.

In addition, we have proposed two second-layer combination schemes, called meta-combination and meta-stacking. This proposal can be considered as the generalization of some special cases in previous studies, for example [Florian & Yarowsky (2002), Florian *et al.* (2002)] used the average and voting rules on the outputs of the first-layer combination rules.

The results of the work on classifier combination are reported in [Le *et al.* (2005a), Le *et al.* (2005b), Le *et al.* (2005c), Le *et al.* (2006a), Le *et al.* (2006c), Le *et al.* (2006d)].

1.3.3 Semi-Supervised Learning

A new bootstrapping algorithm with several variants are generated by providing solutions for the three problems: the problem of imbalance of labeled data, the problem of determining new labeled examples with high confidence, and the problem of generating the final classifier. With the proposed solutions, we have developed new variants for the general bootstrapping algorithm. The experiment has shown that unlabeled data is effective in improving supervised WSD with the proposed bootstrapping algorithms, while this improvement does not appear with the conventional bootstrapping algorithms.

A novel combination model for post semi-supervised learning was proposed. By this model we can combine advantages from classifier combination and semi-supervised learning. Experimental result of this model has reached the state-of-the-art of WSD systems.

This work also implemented a comparison between self-training and co-training with respect to the degree of confidence of new labeled examples. Some discussion was presented. Experimental results showed that self-training give better result than co-training when being used for WSD problem in some particular uses of these algorithms.

The results of this work are reported in [Le *et al.* (2006b), Le *et al.* (2006e)].

1.4 Thesis Structure

This chapter presents an overview of the thesis, including an introduction of word sense disambiguation, the motivations and problems of this thesis, and our contributions. The rest of this thesis is organized as follows.

- Some backgrounds are presented in Chapter 2. First, we present our survey on approaches in WSD including knowledge-based and corpus-based approach. Second, we present three supervised learning algorithms which will be used as basic algorithms in our proposed methods, including Naive Bayes, Support Vector Machines, and Maximum Entropy Models.

- Our work on context representation and feature selection are presented in Chapter 3. This is one of the three main subtasks in this thesis. In this chapter, we first present various kinds of context representation, including results from previous works and our proposal. Next, some new ideas about feature selection for WSD and the corresponding experiments are presented.

- Chapter 4 presents our work on classifier combination, which is one of main contents in our thesis. In this chapter we develop a framework of classifier combination based on Dempster-Shafer (DS) theory of evidence and the notion of Ordered Weighted Averaging (OWA) operators with the help of fuzzy quantifiers. As the result, some combination rules are derived such as DS, max, min, fuzzy majority voting, etc. In addition, we also present two second-layer combination strategies based on meta-combination and the stacking method. These various methods will be experimented with two kinds of individual classifiers, one is based on different feature selection and the other is based on different supervised learning algorithms.

- Chapter 5 presents our work on exploiting unlabeled data to improve performance of supervised WSD (i.e. semi-supervised learning). This work is also a main content of the thesis. We first introduce methods in semi-supervised learning and discuss the reason why we choose the approach in which labeled data is iteratively extended from unlabeled data. Next, we identify the problems that may occur in this approach, and provide solutions for them. As the result, a new semi-supervised learning algorithm with several variants are generated.

- Chapter 6 first summarizes the main points of the thesis including the main achievements and contributions of the thesis, as well as the remaining problems. Finally, several open problems that are in part extended from this thesis will be mentioned in terms of the future research directions.

Chapter 2

Background

There are two parts (sections) in this chapter. In the first part, we will investigate the methods used in WSD which provides an overall picture in abstract level of WSD research. In the second part, we will introduce three supervised learning algorithms, including NB, MEM, and SVM, which will be used as basic algorithms in the machine learning methods of the next chapters.

2.1 Past Research on Word Sense Disambiguation

This section briefly summaries approaches and obtained results in word sense disambiguation studies, specially focusing on current researches. We first present terminologies and describe the task of WSD, and then present the methods which are grouped into rule-based approach and corpus-based approach. Finally, we present the corpora which will be used for experiments in this thesis, and present evaluation metrics of classifiers in WSD.

Given a context containing a polysemous word, the task of WSD is to find the correct sense among potential senses of this word. All disambiguation work involves matching the context of the instance of the word to be disambiguated with either information from an external knowledge (*knowledge-driven* WSD), or information about the contexts of previously disambiguated instances of the word derived from corpora (*data-driven* or *corpus-based* WSD).

2.1.1 Knowledge-based Approach

Work on WSD reached a turning point in the 1980's and 1990's when large-scale lexical resources such as dictionaries, thesauri, and corpora became widely available. Efforts began to attempt to automatically extract knowledge from these sources and to construct large scale knowledge bases by hand.

Machine-readable dictionaries

As the survey in [Ide *et al.* (1998)], Machine-readable dictionaries (MRDs) became a popular source of knowledge for language processing tasks since 1980's. A primary area of activity during 1980's involved attempts to automatically extract lexical and semantic knowledge bases from MRDs. All methods using MRD rely on the notion that the most plausible sense to assign to multiple co-occurring words is the one that maximizes the relatedness among the chosen senses.

[Lesk (1986)] created a knowledge base which associated with each sense in a dictionary a “signature” composed of the list of words appearing in the definition of that sense. Disambiguation was accomplished by selecting the sense of the target word whose signature contained the greatest number of overlaps with the signatures of neighboring words in its context. The 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 very 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 subsequent MRD-based disambiguation work. In some other studies, [Wilks *et al.*(1993)] 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 derive several measures of the degree of relatedness among words. This metric is then used with the help of a vector method that relates each word an its context. [Cowie *et al.* (1992)] used the simulated annealing technique for overcoming the combinatorial explosion of the Lesk’s method.

Inconsistencies in dictionaries are not the only and perhaps not the major source of their limitations for WSD. While dictionaries provide detailed information at the lexical level, they lack pragmatic information that enters into sense determination. For example, the link between *ash* and *tobacco, cigarette*, or *tray* in a network such as Quillian’s is very indirect, whereas in the Brown corpus, the word *ash* co-occurs frequently with one of these words. It is therefore not surprising that corpora have become a primary source of information for WSD.

Thesauri and Computational Lexicons

Thesauri provide information about relationships among words, most notably synonymy. Roget’s International Thesaurus, which was put into machine-tractable form in the 1950’s and has been used in a variety of applications including machine translation, information retrieval, and content analysis, also supplies an explicit concept hierarchy consisting of up to eight increasingly refined levels. Typically, each occurrence of the same word under different categories of the thesaurus represents different senses of that word; i.e., the categories correspond roughly to word senses [Yarowsky (1992)]. A set of words in the same category are semantically related.

[Yarowsky (1992)] derived classes of words by starting with words in common categories in Roget’s (4th edition). A 100–word context of each word in the category is extracted from a corpus (the 1991 electronic text of Grolier’s Encyclopedia), and a mutual information – like statistic is used to identify words most likely to co-occur with the category members. The resulting classes are used to disambiguate new occurrences of a polysemous word: the 100-word context of the polysemous occurrence is examined for words in various classes, and Bayes’ Rule is applied to determine the class most likely to be that of the polysemous word. Since class is assumed by Yarowsky to represent a particular sense of a word, assignment to a class identifies the sense. He reports 92% accuracy on a mean three-way sense distinction. Yarowsky notes that his method is best for extracting topical information, which is in turn most successful for disambiguating nouns (see Section 3.1.2). He uses the broad category distinctions supplied by Roget’s, although he points out that the lower-level information may provide rich information for disambiguation.

Other approaches measure the relatedness between words, taking as a reference a structured semantic net. Thus, [Sussna (1993)] employs the notion of conceptual distance between network nodes in order to improve precision during document indexing. [Agirre & Rigau (1996)] present a method for the resolution of the lexical ambiguity of nouns using the WordNet noun taxonomy and the notion of conceptual density. [Rigau *et al.* (1997)] combine a set of knowledge-based algorithms to accurately disambiguate definitions of MRDs. [Mihalcea & Moldovan (1999)] suggest a method that attempts to disambiguate all the nouns, verbs, adverbs, and adjectives in a given text by referring to the senses provided by WordNet. [Magnini *el al.* (2002)] explore the role of domain information in WSD using WordNet domains; in this case, the underlying hypothesis is that information provided by domain labels offers a natural way to establish semantic relations among word senses, which can be profitably used during the disambiguation process.

Like machine-readable dictionaries, a thesaurus is a resource created for humans and is therefore not a source of perfect information about word relations. It is widely recognized that the upper levels of its concept hierarchy are open to disagreement (although this is certainly true for any concept hierarchy), and that they are so broad as to be of little use in establishing meaningful semantic categories. Nonetheless, thesauri provide a rich network of word associations and a set of semantic categories potentially valuable for language-processing work; however, Roget's and other thesauri have not been used extensively for WSD.

Although knowledge-based systems have been proven to be ready-to-use and scalable tools for all-words WSD because they do not require sense-annotated data, corpus-based systems in general obtain better precision than the knowledge-based ones.

2.1.2 Corpus-based Approach

In the last fifteen years, empirical and statistical approaches have attracted almost studies in NLP field. Many machine learning methods have been applied to a large variety of NLP tasks with remarkable success. The types of NLP problems initially addressed by statistical and machine learning techniques are those of language ambiguity resolution, in which the correct interpretation should be selected from among a set of alternatives in a particular context (e.g., word-choice selection in speech recognition or machine translation, part-of-speech tagging, word-sense disambiguation, co-reference resolution, etc.). These techniques are particularly adequate for NLP because they can be regarded as classification problems, which have been studied extensively in the ML community. Regarding automatic WSD, one of the most successful approaches in the last ten years is supervised learning from examples, in which statistical or ML classification models are induced from semantically annotated corpora. Generally, supervised systems have obtained better results than unsupervised ones, a conclusion that is based on experimental work and international competitions [1]. This approach uses semantically annotated corpora to train machine learning (ML) algorithms to decide which word sense to choose in which contexts. The words in such annotated corpora are tagged manually using semantic classes taken from a particular lexical semantic resource (most commonly WordNet).

[1] http://www.senseval.org

We divide corpus-based methods into three groups including supervised learning approach, unsupervised learning approach, and semi-supervised approach which are corresponding to three types of data used for training: using only annotated data, using only un-annotated data, and using both annotated and un-annotated data. In the following, we will survey WSD studies followed corpus-based approach with respect to these three groups.

Supervised Learning Based Methods

One of the most successful current lines of research is the corpus-based approach, in which statistical or Machine Learning algorithms have been applied to learn statistical models or classifiers from corpora in order to perform WSD. A supervised WSD system requires an annotated dataset which includes labeled (or tagged) examples, in which each example contains the target word **w** assigned with its right sense. This data, called labeled data or training data, is then used for a supervised learning algorithm to train a classifier for future detection of test examples.

Until now, many ML algorithms have been applied, such as: Decision Lists [Yarowsky (1994), Agirre & Martinez (2001)], Neural Networks [Towell & Voorhees (1998)], Bayesian learning [Bruce & Wiebe (1994)], Exemplar-based learning [Ng (1997), Escudero *et al.* (2000a)], Boosting [Escudero *et al.* (2000b)], etc. Further, in [Mooney (1996)] some of the previous methods are compared jointly with Decision Trees and Rule Induction algorithms, on a very restricted domain. Recently, [Lee & Ng(2002)] evaluates some strong ML algorithms in WSD, including Naive Bayes, Support Vector Machines, AdaBoost, and Decision Tree. This work was accomplished for the recently competition data including Senseval-1 and Senseval-2. In the contest of Senseval-3, [Ngai *et al.* (2004)] also investigate the semantic role labeling with Boosting, SVMs, Maximum Entropy, SNOW, and Decision Lists.

Reviewing results from these studies, we can conclude that there is no ML algorithm which dominates others. This conclusion is based on two observations that there are no large distance between the accuracies obtained from different algorithms, and the best algorithm are changed via different corpora.

Classifier Combination Based Methods

As observed in studies of machine learning systems, although one of the available learning systems could be chosen to achieve the best performance for a given pattern recognition problem, the set of patterns misclassified by the different classification systems would not necessarily overlap. This means that different classifiers may potentially offer complementary information about patterns to be classified. This observation highly motivated the recent interest in combining classifiers. Especially, classifier combination for lexical disambiguation in WSD has, not surprisingly, received much attention recently from the community.

In the WSD literature, the first empirical study of combining classifiers was presented in [Kilgarriff & Rosenzweig (2000)], in which the authors combined the output of the participating SENSEVAL1 systems via simple voting. [Pedersen (2000)] built an ensemble of Naive Bayesian classifiers, each of which is based on lexical features that represent co-occurring words in varying sized windows of context. [Klein *et al.* (2002)] use a stacking type of combination techniques with some combination strategies including majority voting, weighted voting. [Hoste *et al.* (2002)] used word experts consisting of four memory-based learners trained on different context. Output of the word experts is based on majority voting or weighted voting. In [Florian *et al.* (2002)], the authors used six different classifiers as components of their combination. They compared several different combination strategies which include combining the posterior distribution, combination based on order statistics, and several different voting strategies. [Wang & Matsumoto (2004)] presented a kind of stacking; individual classifiers were built using NB with varying sized windows of context that are similar to Pedersen's approach [Pedersen (2000)], and then used K-nearest neighbors as the meta learning method.

Unsupervised Learning Based Methods

Although supervised methods typically achieve better performance than unsupervised alternatives, their applicability is limited to those words for which sense labeled data exists, and their accuracy is strongly correlated with the amount of labeled data available [Yarowsky *el al.* (2002)]. Furthermore, obtaining manually labeled corpora with word senses is costly and the task must be repeated for new domains, languages, or sense inventories. There are not many studies about unsupervised WSD, that is easy to understand because unsupervised WSD obtains lower accuracy in comparison with supervised learning. However, it is very difficulty to achieve a large annotated corpora for all ambiguous words, which is prerequisite for building a good supervised WSD system. In that case, an unsupervised WSD is necessary. Moreover, this work is useful not only for preparing annotated corpora or constructing dictionary, but also for some other NLP tasks such as Information Retrieval, (see [Schutze (1998)] for an example).

The essential problem in unsupervised learning for WSD is how to group the given contexts of an ambiguous word into cluster such that the contexts in the same cluster have the same sense of this ambiguous word. The most popular work is [Schutze (1998)], in which senses are represented in Word Space and the EM algorithm are invoked to optimize the parameters of the obtained classifiers. Recently, Pedersen and his co-workers have tried to build an WSD system and their results are published in [Purandare & Pedersen (2004), Kulkarni & Pedersen (2005)]. [Brody *et al.* (2006)] used ensemble approach to unsupervised WSD and based on predominant senses which are derived automatically from raw text.

Semi-Supervised Learning Based Methods

Due to the difficulty of obtaining labeled data, while unlabeled data is abundant and cheap to collect, recently several WSD studies have tried to use unlabeled data to boost the performance of supervised learning. The process of using both labeled and unlabeled data to build a classifier is called *semi-supervised learning*. With a small number of labeled examples, we may face two problems: the first one is that we cannot archive the correct probability distribution over feature space; the second one is that there is lack of features in the training data, i.e. we may see new features in test data. There were several studies which aimed to overcome the second problem by using external resources as, e.g., thesaurus or lexicons to disambiguate word senses or automatically generate sense-tagged corpus such as in [Lesk (1986), Lin (1997), McCarthy *et al.* (2004), Seo *et al.* (2004), Yarowsky (1992)] or by using similarity between words based on untagged corpus such as in [Karov & Edelman (1998)], or by basing on the WordNet resource such as in [Leacock (1998)]. There is also another approach in the case of lacking labeled sense-tagged examples, in which word senses are distinguished with the help of a second language. In this approach, the studies exploited the differences between mapping of words to senses in different languages making use of bilingual corpora (e.g. parallel corpora or untagged monolingual corpora in two languages), such as in [Brown *et al.* (1991), Dagan & Itai (1994), Diab & Resnik (2002), Li & Li, Ng *et al.* (2003)].

Recently, almost related studies interest in using semi-supervised learning algorithms to extend labeled data. Among them, Yarowsky algorithm in [Yarowsky (1995)] can be considered as the first bootstrapping algorithm. In this algorithm, the author used some labeled examples as seeds and extracted from them the decision rules, which are then used to detect senses for new examples. This algorithm was based on the principle "one sense per collocation". As the result, new labeled examples were obtained and added to the current labeled dataset. New decision rules were continuously extracted and this process was repeated until converged. [Mihalcea (2004)] did an investigation of application of co-training and self-training to word sense disambiguation. In another study, [Zheng (2005)] applied the Label Propagation based semi-supervised learning algorithm proposed by [Zhu & Ghahramani (2002)] to WSD. They also suggested an entropy-based method to automatically identify a distance measure to boost the performance of Label Propagation algorithm on a given dataset. [Pham *et al.* (2005)] applied an algorithm which used co-training in spectral graph transductive(SGT) [Joachims (2003)] for WSD. Instead of directly computing the nearest neighbor graph in SGT, the authors constructed a separate graph for each view, and combined them together to obtain the final graph.

2.1.3 Corpora and Evaluation

Corpora

In the scope of this thesis, we just consider the corpora which aims to serve for corpus-base approach and related for only lexical task. The following are some corpora wildly used in WSD community and will be used in this thesis.

– Datasets of the four words, namely *interest*, *line*, *serve*, and *hard*, which are used in numerous comparative studies of word sense disambiguation methodologies [Pedersen (2000), Leacock (1998), Ng & Lee (1996), Bruce & Wiebe (1994)]. There are 2369 instances of *interest* with 6 senses, 4143 instances of *line* with 6 senses, 4378 instances of *serve* with 4 senses, and 4342 instances of *hard* with 3 senses.

– In 1998, the first contest for word sense disambiguation was organized, and called SENSEVAL-1. Next, the second and the third contests took place in 2001 and 2004 respectively. Their data for the competition are called Senseval-1, Senseval-2, and Senseval-3 respectively, and quickly became the most popular corpora in the community. There are 53 lexical item, 73 lexical items, and 57 lexical items in English lexical sample of Senseval-1, Senseval-2, and Senseval-3, respectively. Data for each lexical item contains a training dataset and a test dataset. Note that in each instance (or example) in Senseval's data the target word may be assigned with multiple senses. For an example which are assigned with K senses ($K > 1$), we multiple them with K instances, each instance is assigned with one sense in these K senses.

Table 2.1: Experimental Results on Senseval-2 from Previous Studies

Method	Accuracy
ASO multi-task, [Ando(2006)]	*68.1* (optimized parameter)
classifier combination, [Florian & Yarowsky (2002)]	66.5
polynomial KPCA, [Wu *et al.* (2004)]	65.8
SVM, [Lee & Ng(2002)]	65.4
Senseval-2 best system	64.2

Table 2.2: Experimental Results on Senseval-3 from Previous Studies

Method	Accuracy
ASO: multi-task learning	73.8
ASO: semi-supervised learning	73.5
ASO: muli-task + semi-supervised learning	74.1
Senseval-3 best system	72.9

Evaluation Metrics

Word sense disambiguation classifiers are usually evaluated using a standard measure metric in Machine Learning: the prediction accuracy (or *accuracy* for short).

Assume we have a WSD classifier H and a test data consisting of N labeled examples which are labeled by humans. Consider these examples as unlabeled ones and using H to detect labels (senses) for these N examples, and suppose that among there are m examples correctly label detected. Then the prediction accuracy of H on the test data is define as follows:

$$\begin{aligned} accuracy &= \frac{\text{the number of \textit{correctly} detected labels predicted by the classifier}}{\text{the size of the test data}} \\ &= \frac{m}{N} \end{aligned}$$

Table 2.1 and Table 2.2 show experimental results on Senseval-2 and Senseval-3, respectively, from the best WSD systems of the current researches.

2.2 Supervised Learning Algorithms

In this section we briefly introduce three supervised learning algorithms which will be used as the supervised basic algorithms in our investigations and proposals for feature selection, classifier combination, and exploiting unlabeled data. These algorithms include Naive Bayes, Maximum Entropy Models, and Support Vector Machines which are very popular and strong supervised learning algorithms in many classification problems such as text categorization, part-of-speech tagging, etc. and of course effective for word sense disambiguation.

In all related studies, like many problems in natural language processing, word sense disambiguation is considered as a classification problem in which the task if to observe some linguistic "context" $b \in \mathcal{B}$ and predict the correct linguistic "class" $a \in \mathcal{A}$. This involves constructing a classifier $h : \mathcal{B} \rightarrow \mathcal{A}$, which in turn can be implemented with a conditional probability distribution p, such that $p(a|b)$ is the probability of "class" a given some "context" b. For WSD problem, the context b may consist of some evidences (features) such word, collocation, part-of-speech tags, etc. Large text corpora usually contain some information about the co-occurrence of $a's$ and $b's$, but never enough to reliably specify $p(a|b)$ for all possible (a, b) pairs. The challenge is then to find a method for using the partial evidence about the $a's$ and $b's$ to reliably estimate the probability model p.

Below, I will introduce in order the classification models which are based on maximum entropy model, support vector machines, and Naive Bayesian rule.

2.2.1 Maximum Entropy Models

Learning with Maximum Likelihood Estimation on Exponential Models

The work in this thesis fits in what is called the corpus-based approach. In this approach, we assume the existance of a training set $\mathcal{T} = \{(a_1, b_1), \ldots, (a_N, b_N)\}$, which is a large set of contexts $b_1, \ldots, b_N$ that have been annotated with their correct classes $a_1, \ldots$

One way to combine evidence is to "weight" the features by using them in a log-linear, or exponential, model:

$$p(a|b) = \frac{1}{Z(b)} \prod_{j=1}^{k} \alpha_j^{f_j(a,b)} \tag{2.1}$$

$$Z(b) = \sum_a \prod_{j=1}^{k} \alpha_j^{f_j(a,b)}$$

where k is the number of features and $Z(b)$ is a normalization factor to ensure that $\sum_a p(a|b) = 1$. Each parameter α_j, where $\alpha_j > 0$, corresponds to one feature f_j and can be interpreted as a "weight" for that feature. The probability $p(a|b)$ is then a normalized product of those feature that are "active" on the (a, b) pair, i.e., those feature f_j such that $f_j(a, b) = 1$. The weights $\alpha_1, \ldots, \alpha_k$ of the probability distribution p^* that best fit the training data can be obtained with the popular technique of maximum likelihood estimation:

$$\mathcal{Q} = \{p | p(a|b) = \frac{1}{Z(b)} \prod_{j=1}^{k} \alpha_j^{f_j(a,b)}\}$$

$$L(p) = \sum_{a,b} \tilde{p}(a, b) \text{log} p(a, b)$$

$$p^* = \arg \max_{q \in \mathcal{Q}} L(q)$$

where $\mathcal{Q}$ is the set of models of log-linear form, $\tilde{p}(a,b)$ is the probability of seeing (a,b) in the training set $\mathcal{T}$, $L(p)$ is the conditional log-likelihood of the training set $\mathcal{T}$, and p^* is the optimal probability distribution according to the maximum likelihood criterion.

Conditional Maximum Entropy Models

As mention before, suppose that we have k features and given a linguistic prediction $a \in \mathcal{A}$ and an observable context $b \in \mathcal{B}$, the ultimate goal is to find an estimate for the conditional probability $p(a|b)$. In the conditional maximum entropy framework used in [Berger *et al.* (1996)], the optimal solution p^* is the most uncertain distribution that satisfies the k constraints of feature expectations:

$$\begin{aligned}
p^* &= \arg\max_{p \in P} H(p) \\
H(p) &= -\sum_{a,b} \tilde{p}(b)p(a|b)\log p(a|b) \\
P &= \{p|E_p f_j = E_{\tilde{p}} f_j,\ j = 1, \ldots, k\} \\
E_{\tilde{p}} f_j &= \sum_{a,b} \tilde{p}(a,b) f_j(a,b) \\
E_p f_j &= \sum_{a,b} \tilde{p}(b)p(a|b) f_j(a,b)
\end{aligned}$$

An important difference here from the simple example is that $H(p)$ denotes the conditional entropy averaged over the training set, as opposed to the joint entropy, and that the marginal probability of b used here is the observed probability $\tilde{p}(b)$, as opposed to a model probability $p(b)$. As presented in [Berger *et al.* (1996)], $\tilde{p}(b)$ is considered as a marginal probability. Here $E_p f_j$ is the model $p's$ expectation of f_j, $E_{\tilde{p}} f_j$ denotes the observed expectation of a feature f_j, $\tilde{p}(a,b)$ denotes the observed probability of (a,b) in some fixed training sample, and P denotes the set of probability models that are consistent with the observed evidence.

Relationship to Maximum Likelihood

In general, the maximum likelihood and maximum entropy frame works are two different approaches to statistical modelling, but in this case they yield the same answer. It can show that maximum likelihood parameter estimation for models of form 2.1 is equivalent to maximum entropy parameter estimation over the set of consistent models. That is,

$$p^* = \arg\max_{q \in \mathcal{Q}} L(q) = \arg\max_{p \in \mathcal{P}} H(p)$$

Under the maximum likelihood criterion, p^* will fit the data as closely as possible, while under the maximum entropy criterion, p^* will not assume anything beyond the information in the linear constraints that define P. A proof in [Ratnaparkhi (1998)] shows that the condition $p^* = \arg\max_{q \in \mathcal{Q}} L(q)$ is equivalent to the condition that $p^* = \arg\max_{p \in \mathcal{P}} H(p)$.

Parameter Estimation

In [Ratnaparkhi (1998)], the author uses an algorithm called *Generalized Iterative Scaling*, or GIS, to find values for the parameter of p^*. The algorithm is presented in the procedure called GIS as follows

$$\alpha_j^{(0)} = 1$$
$$\alpha_j^{(n+1)} = \alpha_j^{(n)} [\frac{E_{\tilde{p}} f_j}{E_{p^{(n)}} f_j}]^{\frac{1}{C}}$$

where C is some constant and

$$E_{p^{(n)}} f_j = \sum_{a,b} \tilde{p}(b) p^{(n)}(a|b) f_j(a,b)$$
$$p^{(n)}(a|b) = \frac{1}{Z(b)} \prod_{j=1}^{l} (\alpha_j^{(n)})^{f_j(a,b)}$$

Given k features, the GIS procedure requires computation of each observed expectation $E_{\tilde{p}} f_j$, and requires re-computation of the model's expectation $E_p f_j$ on each iteration, for $j = 1, \ldots, k$. The quantity $E_{\tilde{p}} f_j$ is merely the count of f_j normalized over the training set:

$$E_{\tilde{p}} f_j = \sum_{a,b} \tilde{p}(a,b) f_j(a,b) = \frac{1}{N} \sum_{i=1}^{N} f_j(a_i, b_i)$$

where N is size of the training set $\mathcal{T} = \{(a_1, b_1), \ldots, (a_N, b_N)\}$.

The computation of $E_p f_j$ involves summing over each context b in the training set, and each $a \in \mathcal{A}$:

$$E_{p^{(n)}} f_j = \sum_{a,b} \tilde{p}(b) p^{(n)}(a|b) f_j(a,b)$$

Note that $f_j(a_i, b_i)$ can be simply a binary function, defined as follows:

$$f_j(a_i, b_i) = \begin{cases} 1, & \text{if } b_i\text{``}\mathit{contain}\text{''} f_j \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

2.2.2 Support Vector Machines

Support Vector Machines (SVMs) are based on the principle of Structural Risk Minimization from the Statistical Learning Theory [Vapnik (1998)] and, in their basic form, they learn a linear discriminant that separates a set of positive examples from a set of negative examples with maximum margin (the margin is defined by the distance of the hyperplane to the nearest of the positive and negative examples). This learning bias has proved to have good properties in terms of generalization bounds for the induced classifiers.

Linear SVM

SVM (Support Vector Machine) is a useful technique for data classification. A classification task usually involves with training and testing data which consist of some data instances. Suppose we have N training data points $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$ where $\mathbf{x}_i \in \mathcal{R}^d$ and $y_i \in \{\pm 1\}$. We would like to learn a linear separating hyperplane classifier:

$$f(x) = sgn(\mathbf{w}.\mathbf{x} - b)$$

Furthermore, we want this hyperplane to have the maximum separating margin with respect to the two classes. Specifically, we want to find this hyperplane $H : y = \mathbf{w}.\mathbf{x} - b = 0$ and two hyper-planes parallel to it and with equal distance to it,

$$H_1 : y = \mathbf{w}.\mathbf{x} - b = +1$$
$$H_2 : y = \mathbf{w}.\mathbf{x} - b = -1$$

with the condition that there are no data points between H_1 and H_2, and the distance between H_1 and H_2 is maximized.

For any separating plane H and the corresponding H_1 and H_2, we can always normalize the coefficients vector so that H_1 will be $y = \mathbf{w}.\mathbf{x} - b = +1$ and H_2 will be $y = \mathbf{w}.\mathbf{x} - b = -1$.

We want to maximize the distance between H_1 and H_2. So there will be some positive examples on H_1 and some negative examples on H_2. These examples are called *support vectors* because only they participate in the definition of the separating hyperplane, and other examples can be removed and/or moved around as long as they do not cross the planes H_1 and H„.

Recall that in 2-D, the distance from a point $(\mathbf{x}_0, y_0)$ to a line $Ax + By + C = 0$ is

$$\frac{|A\mathbf{x}_0 + By_0 + C|}{\sqrt{A^2 + B^2}}$$

Similarly, the distance between H_1 and H_2 is $\frac{2}{\|w\|}$. So, in order to maximize the distance, we should minimize $\|\mathbf{w}\| = \mathbf{w}^T\mathbf{w}$ with the condition that there are no data points between H_1 and H_2:

$$\mathbf{w}.\mathbf{x} - b \geq +1, \text{ for positive examples } y_i = +1$$
$$\mathbf{w}.\mathbf{x} - b \leq -1 \text{ for negative examples } y_i = -1$$

These two conditions can be combined into

$$y_i(\mathbf{w}.\mathbf{x}_i - b) \geq 1$$

So the problem can be formulated as

$$\min_{\mathbf{w},b} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$
$$\text{subject to } y_i(\mathbf{w}.\mathbf{x}_i - b) \geq 1$$

This is a convex, quadratic programming problem (in $mathbfw, b$), in a convex set. Introducing Lagrange multipliers $\alpha_1, \ldots, \alpha_N \geq 0$ we have the following Lagrangian:

$$\mathcal{L}(\mathbf{w}, b, \alpha) \equiv \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i (\mathbf{w}.\mathbf{x}_i - b) + \sum_{i=1}^{N} \alpha_i$$

The Dual Problem

We can solve the Wolfe dual instead: maximize $\mathcal{L}(\mathbf{w}, b, \alpha)$ with respect to α, subject to the constraints that the gradient of $\mathcal{L}(\mathbf{w}, b, \alpha)$ with respect to the primal variables $\mathbf{w}$ and b vanish:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{w}} &= 0 \\ \frac{\partial \mathcal{L}}{\partial b} &= 0 \\ \text{and that } \alpha &\geq 0 \end{aligned}$$

Form these equations, we have

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \\ \sum_{i=1}^{N} \alpha_i y_i &= 0 \end{aligned}$$

Substitute them into $\mathcal{L}(\mathbf{w}, b, \alpha)$, we have

$$\mathcal{L} \equiv \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i \mathbf{x}_j$$

in which the primal variables are eliminated.

When we solve α_i, we can get $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$, and we can classify a new object $\mathbf{x}$ with

$$\begin{aligned} f(\mathbf{x}) &= sgn(\mathbf{w}.\mathbf{x} + b) \\ &= sgn((\sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i).\mathbf{x} + b) \qquad (2.3) \\ &= sgn(\sum_{i=1}^{N} \alpha_i y_i (\mathbf{x}_i.\mathbf{x}) + b) \qquad (2.4) \\ & \qquad (2.5) \end{aligned}$$

Please note that in the objective function and the solution, the training vectors $\mathbf{x}_i$ occur aoly in the form of dot product.

Non-linear SVM

If the surface separating the two classes are not linear, we can transform the data points to another high dimensional sapce such that the data points will be linearly separable. Let the transformation be $\Phi(.)$. In the high dimensional space, we solve

$$\mathcal{L}_D \equiv \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \Phi(\mathbf{x_i}) \Phi(\mathbf{x_j})$$

Suppose, in addition, $\Phi(\mathbf{x_i})\Phi(\mathbf{x_j} = k(\mathbf{x_i}, \mathbf{x_j})$. That is the dot product in that high dimensional space is equivalent to a *kernel* function of the input space. So we need not be explicit about the transformation $\Phi(.)$ as long as we know that the kernel function $k(\mathbf{x_i}, \mathbf{x_j})$ is equivalent to the dot product of some other high dimensional space. There are many kernel functions that can be used this way, for example:

- Linear: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- Polynomial:$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i^T \mathbf{x}_j + r)^d, \ \gamma > 0$
- Radial basis function (RBF): $K(\mathbf{x}_i, \mathbf{x}_j) = exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2), \ \gamma > 0$
- Sigmoid: $K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i^T \mathbf{x}_j + r)$

Here, γ, r, and d are kernel parameters.

2.2.3 Naive Bayes

Naive Bayes is the simplest representative of probabilistic learning methods. Event that it is shown to be quite effective for WSD task, such as in [Lee & Ng(2002)]. Moreover, it is also used as basic supervised learning algorithm for other learning approaches than supervised learning, such as classifier combination [Hoste *et al.* (2002), Pedersen (2000), Klein *et al.* (2002)] or in our work [Le *et al.* (2005a), Le *et al.* (2005b), Le *et al.* (2006a)] and semi-supervised learning [Pham *et al.* (2005), Mihalcea (2004)].

We assume that the polysemous word w is disambiguating. Suppose w has a set of potential senses (classes) $\omega = \{\omega_1, \ldots, \omega_c\}$, and given a context of w which is presented by a set of features $\mathbf{f} = \{f_1, \ldots, f_n\}$. The Bayesian theory suggests that the word w should be assigned to class ω_k provided the a posteriori probability of that class is maximum, namely

$$k = \arg\max_j P(\omega_j | \mathbf{f})$$

According to Bayes theory, we have:

$$P(\omega_j | \mathbf{f}) = \frac{P(\omega_j) P(\mathbf{f} | \omega_j)}{P(\mathbf{f})} = \frac{P(\omega_j) \prod_{i=1}^{n} P(f_i | \omega_j)}{P(\mathbf{f})} \tag{2.6}$$

Let us denote

$$r_j = \frac{P(\omega_j | \mathbf{f})}{P(\omega_1 | \mathbf{f})}, \ \text{for } j = 1, \ldots, c$$

With this notation, we immediately obtain

$$P(\omega_1|\mathbf{f}) = \frac{1}{\sum_{j=1}^{c} r_j} \tag{2.7}$$

Clearly, $r_1 = 1$. We will then compute r_j ($j = 2, \ldots, c$) based on the following formulation. From (2.6), we have

$$r_j = \frac{P(\omega_j|\mathbf{f})}{P(\omega_1|\mathbf{f})} = \frac{P(\omega_j)\prod_{i=1}^{n} P(f_i|\omega_j)}{P(\omega_1)\prod_{i=1}^{n} P(f_i|\omega_1)}$$

Taking the log of the last expression, we obtain

$$\log(r_j) = \sum_{i=1}^{n} \log(P(f_i|\omega_j)) + \log(P(\omega_j)) - \sum_{i=1}^{n} \log(P(f_i|\omega_1)) - \log(P(\omega_1)) \tag{2.8}$$

which is easy to compute more exactly. Once all r_j are computed via (2.8), it is easily to derive probabilities $P(\omega_j|\mathbf{f})$, for $j = 1, \ldots, c$, from (2.7).

The probability of sense ω_j, $P(\omega_j)$, and the conditional probability of a feature f_i given the sense ω_j, $P(f_i|\omega_j)$, are computed via maximum-likelihood estimation as:

$$P(\omega_j) = \frac{\text{count}(\omega_j)}{N}$$

and

$$P(f_i|\omega_j) = \frac{\text{count}(f_i, \omega_j)}{\text{count}(\omega_j)}$$

where $\text{count}(f_i, \omega_j)$ is the number of occurrences of f_i in a context of sense ω_j in the training corpus, $\text{count}(\omega_j)$ is the number of occurrences of ω_j in the training corpus, and N is the total number of occurrences of the polysemous word w or the size of the training dataset. To avoid the effects of zero counts when estimating the conditional probabilities of the model, when meeting a new feature f_i in a context of the test dataset, for each sense ω_j we set $P(f_i|\omega_j)$ equal to $\frac{1}{N}$.

2.3 Summary

The first part of this chapter has briefly investigated methods used in word sense disambiguation and shown an overall picture in abstract level of WSD study. Some concepts, data, and evaluation, which are preparation steps for our work, are also presented. Through this investigation, we can see that the accuracy of all past studies is around 70%, so the WSD task is still an open problem. Moreover, thought various aspects of machine learning have been applied to WSD, it is necessary to discover new aspects of machine learning and to find out whether it is helpful for improving performance of WSD systems. Particularly, from the view of machine learning, we see that the two approaches including classifier combination and unlabeled data exploitation for WSD are just researched at an initial step, so that only some aspects of these approaches are investigated with limited results. This observation motivates us to focus our study on these approach. In addition, feature selection is an important factor to achieve high performance of WSD systems, so that it is also a main work in this thesis.

In the second part, we have briefly introduce three supervised learning algorithms, including NB, MEM, and SVM, which will be used as basic algorithms for the learning approaches in our works: feature selection, classifier combination, and semi-supervised learning. Among these algorithms the NB based classifier is built by ourself, and the MEM based classifier and SVM based classifier are borrowed from published tools. This is the reason why we do not go deeper in building MEM and SVM classifiers. In particular, MEM library is borrowed from $http : //www - tsujii.is.s.u - tokyo.ac.jp/\ tsuruoka/maxent/$ and SVM library is borrowed in LIBSVM $http : //www.csie.ntu.edu.tw/\ cjlin/libsvm/$. It is also worth to note that this SVM tool supports for multi-class problem so it is convenient for us to use in our experiment. We also use the default parameters of SVM library with linear kernel because through trying some parameter setting we see that it gives the best accuracy.

Chapter 3

Context Representation and Feature Selection

One of the most important tasks in WSD is the determination of useful information for determining word senses. In the corpus-based approach, most studies just consider the information extracted from the context in which the target word appears. From our observation and other investigation such as [Klein *et al.* (2002), Mihalcea (2004)], we see that designing features plays an important role for classifiers to obtain high accuracy. Further, we observed that, in the context of WSD, the choice of what method of feature selection to use may more strongly influence the quality of classifiers than the choice of what machine learning algorithm to apply.

In this chapter, we consider the work of information determination in two aspects: context representation and feature selection. The first aspect involves selecting useful knowledge sources: we first consider various knowledge sources and try to answer the question that combine all these kinds or only some of them, which is the best? The latter aspect involves the selection of individual features to which we will find out whether some features can be considered to be redundant or noised to remove from the whole set of features.

Therefore, in this chapter we first investigate various knowledge sources (i.e. context representations) and represent them as features. And then, we present proposed methods for feature selection. Note that, to the task of feature selection we divide it into two subtasks: selection of knowledge sources (which are represented by feature subsets) and selection of individual features.

3.1 Context Representation

3.1.1 The Kinds of Knowledge

Context is the only means to identify the meaning of a polysemous word. Therefore, all work on sense disambiguation relies on the context of the target word to provide information to be used for its disambiguation. For corpus-based methods, context also provides the prior knowledge with which current context is compared to achieve disambiguation.

According to [Ide *et al.* (1998)], context is used in two ways:

- *The bag of words*: here, context is considered as words in some window surrounding the target word, regarded as a group without consideration for their relationship to the target in terms of distance, grammatical relations, ect.

- *Relational information*: context is considered in terms of some relations, selectional preferences, orthographic properties, phrasal collocation, semantic categories, ect.

[Ng & Lee (1996)] is considered as the first study in which various linguistic knowledge sources are used, including topical context, collocation of words, and the verb-object syntactic relationship. [Leacock (1998)] used more kinds of information, that are words or part-of-speech tags assigned with their positions in relation to the target word. More currently, [Lee & Ng(2002)] used all these kinds of information, which then become popular sources of knowledge for recent studies, but with some modifications (add or remove some knowledge sources), se [Le & Shimazu (2004), Le *et al.* (2005a), Ando(2006)].

Suppose that $\mathbf{w}$ is the polysemous word to be disambiguated, and $S = \{s_1, \ldots, s_m\}$ is the set of its potential senses. Given a context W of $\mathbf{w}$ represented as:

$$W = w_{-N^L}, \ldots, w_{-1}, w_0, w_{+1}, \ldots, w_{+N^R} \tag{3.1}$$

W is a context of $\mathbf{w}$ within a windows $(-N^L, +N^R)$ in which $w_0 = \mathbf{w}$ is the target word; for each $i \in \{N^L, \ldots, -1, +1, \ldots, +N^R\}$, w_i is a word appearing at the position i in relation with $\mathbf{w}$. If $i < 0$ then w_i stands in the left of $\mathbf{w}$, and if $i > 0$ then w_i stands in the right of $\mathbf{w}$. For simplify, we assume $N^L = N^R$ and denote this value by N.

Up to now, most studies use part-of-speech information as an important knowledge source for determining word senses. Therefore, the sentences containing $\mathbf{w}$ should be POS tagged. Denote the context in (3.1) after POS tagging as:

$$W = [w_{-N}, p_{-N}], \ldots, [w_{-1}, p_{-1}], [w_0, p_0], [w_{+1}, p_{+1}], \ldots, [w_{+N}, p_{+N}] \tag{3.2}$$

In the below, we divide the usually used knowledge into the four kinds.

Topical Context

Topical context includes substantive words that co-occur with a given sense of the polysemous word, usually within a window of several sentences. Unlike micro-context, which has played a role in disambiguation work since the early 1950s, topical context has been less consistently used. Methods relying on topical context exploit redundancy in a text–that is, the repeated use of words that are semantically related throughout a text on a given topic. Thus, base is ambiguous, but its appearance in a document containing words such as *pitcher*, and *ball* is likely to isolate a given sense for that word (as well as the others, which are also ambiguous). Work involving topical context typically uses the bag-of-words approach, in which words in the context are regarded as an unordered set. [Yarowsky (1992)] uses a 100-word window, both to derive classes of related words and as context surrounding the polysemous target, in his experiments using Roget's Thesaurus. [Gale *et al.* (1993)], looking at a context of ±50 words, indicate that while words closest to the target contribute most to disambiguation, they improved their results from 86% to 90% by expanding context from ±6 (a typical span when only micro-context is considered) to ±50 words around the target. All studies use this kind of information as an important part of the whole knowledge used for disambiguating senses, such as [Pedersen (2000), Lee & Ng(2002)].

Topical context is represented by a set of unordered words in a certain window size. Particularly, if the context is represented as in (3.2), then a topic context in a window $(-M, +M)$ is represented by the set, denoted by TC, as follows:

$$TC = \{w_{-M}, \ldots, w_{-1}, w_{+1}, \ldots, w_{+M}\}$$

Local Words

Using "Local Words" we want to mention the information extracted from "the words in a local context". Note that a "local context" is a context containing the target word in a small size. In our opinion, *collocations* and *ordered words*, which are wildly used in WSD studies, can be grouped into this kind of information.

Collocations According to [Ide *et al.* (1998)], a significant collocation can be defined as a syntagmatic association among lexical items. With the context W as represented in (3.2), a collocation is defined as a sequence of words from the position $-l$ to the position $+r$: $w_{-l} \ldots w_0 \ldots w_r$, where $l \geq 0$, $r \geq 0$, and $l + r \geq 1$. As usually used in previous works, we design a set of collocations based on the maximum length of these collocations. Denote $ColW$ be the set of collocations of maximum length Len, it is defined as:

$$ColW = \{w_{-l} \ldots w_0 \ldots w_{+r} | l \geq 0; r \geq 0; l + r \geq 1; l + r \leq Len\}$$

Ordered Words Different to unordered words in topical context, each ordered word consists of a word and its position in relationship with the target word. In our view, ordered words in a local context contain information about semantic and syntax relations between neighbor words and the target word. The set of ordered words in a local window (-l,+r), denoted by OW, consists of pairs (w_i, i), as follows.

$$OW = \{(w_i, i) | i = -l, \ldots, +r\}$$

Local Part-Of-Speeches

Using "Local Part-Of-Speeches (POSs)" we want to mention the information extracted from "the Part-Of-Speeches tags in a local context". Similar to Local Words, the kinds of information of also consists of collocations of POSs and ordered POSs. The difference here is that the Local Word involves the orthographic forms of the neighbor words while Local POS involves their part-of-speech forms. In a window $(-l, +r)$ the set of collocation of POSs with maximum length Len, denoted by $ColP$, and the set of ordered POSs, denoted by OP, are formed as:

$$ColP = \{p_{-l} \ldots w_0 \ldots p_{+r} | l \geq 0; r \geq 0; l + r \geq 1; l + r \leq Len\}$$

$$OP = \{(p_i, i) | i = -l, \ldots, r\}$$

Syntactic Relations

[Hearst (1991)] segments text into noun phrases, prepositional phrases, and verb groups, and discards all other syntactic information. [Yarowsky (1993)] determines various behaviors based on syntactic category; for example, that verbs derive more disambiguating information from their objects than from their subjects, adjectives derive almost all disambiguating information from the nouns they modify, and nouns are best disambiguated by directly adjacent adjectives or nouns. In other works, syntactic information most often is simply part of speech, used invariably in conjunction with other kinds of information such as [Bruce & Wiebe (1994), Leacock (1998)]. Evidence suggests that different kinds of disambiguation procedures are needed depending on the syntactic category and other characteristics of the target word [Yarowsky (1993), Leacock (1998)]. Most recent studies also use syntactic information such as in [Lee & Ng(2002), Montoyo *et al.* (2005), Ando(2006)]. However, there is no an unique use of syntactic information through all these works. This circumstance can be seen, at least, in two aspects: the used syntactic parser, and the syntactic relations. For example, [Hearst (1991)] have avoided complex processing by using shallow parsing to achieve noun phrases, prepositional phrases, and verb groups, and then extract from these phrases the complementary components of the target word as the syntactic information; [Lee & Ng(2002)] parsed sentences containing the target word using a statistical parser in [Charniak (2000)], and then the generated constituent tree is converted into a dependency tree to obtain syntactic information; [Ando(2006)] used the Slot Grammar-based full parser ESG in [McCord (1990)] and extracted several syntactic relations such as subject-of, object-of, and noun modifier.

3.1.2 Our Selection

In our work, we use all kinds of knowledge as mentioned above and represent them as subsets of features, as below:

- *bag-of-words*, $F_1(l, r) = \{w_{-l}, \ldots, w_{+r}\}$: We investigate three sets of this knowledge including $F_1^a = F_1(-5, +5)$, $F_1^b = F_1(-10, +10)$, $F_1^c = F_1(-100, +100)$, corresponding to small size, medium size, and large size, respectively.

- *collocation of words*, $F_2 = \{w_{-l} \ldots w_{+r}\}$: As a result of our work in [Le & Shimazu (2004)] we choose collocations such that their lengths (including the target word) are less or equal to 4, it means $(l + r + 1) \leq 4$.

- *ordered words*, $F_3 = \{w_i | i = -l, \ldots, +r\}$: We choose $l = r = 3$

- *collocation of POSs*, $F_4 = \{p_{-l} \ldots p_{+r}\}$: Like collocation of words, we choose their lengths including the target word) are less or equal to 4.

- *ordered POSs*: $F_5 = \{p_i | i = -l, \ldots, +r\}$: We choose $l = r = 3$

- *Syntactic features*, F_6: in order to extract syntactic information, we used a shallow parsing obtained from [Tsuruoka & Tsujii (2005)]. From the output of this parser, we select verb-phrases and noun-phrases which contain the target word, and then extract from them the syntactic relations, such as verb-noun, adjective-noun, verb-adjective. For examples, a context of the target word *serve* is chunked (shallow parsed) as follows, *(WHNP (WP what)) (NP (NNS flights)) (VBP are) (RB there) (IN from) (NP (NNP nashville)) (PP (TO to) (NP (NNP houston))) (NP (NN tomorrow) (NN evening)) (SBAR (WHNP (WDT that)) (S (VP (VB* ***serve****) (NP (NN dinner)))))*

and then, we can extract a *verb-noun* relation, that is *serve-dinner*.

For more detail, we extract syntactic features as follows: Each syntactic feature is represented as a pair of the target word and a content word corresponding to a syntactic relation such as *verb-noun*, or *adjective-noun*, or *verb-adjective*. To obtain such features, we first collect verb phrases (VPs) and noun phrases (NPs) which contain the target word. Note that there may be more than one of VPs (or NPs), and in such case we select the phrase with the shortest length. Then, each content word (it must be noun, verb, or adjective and is not the target word) in the selected phrases will be combined with the target word to generate a pair of words. This pair in the corresponding syntactic relation is considered as a syntactic feature.

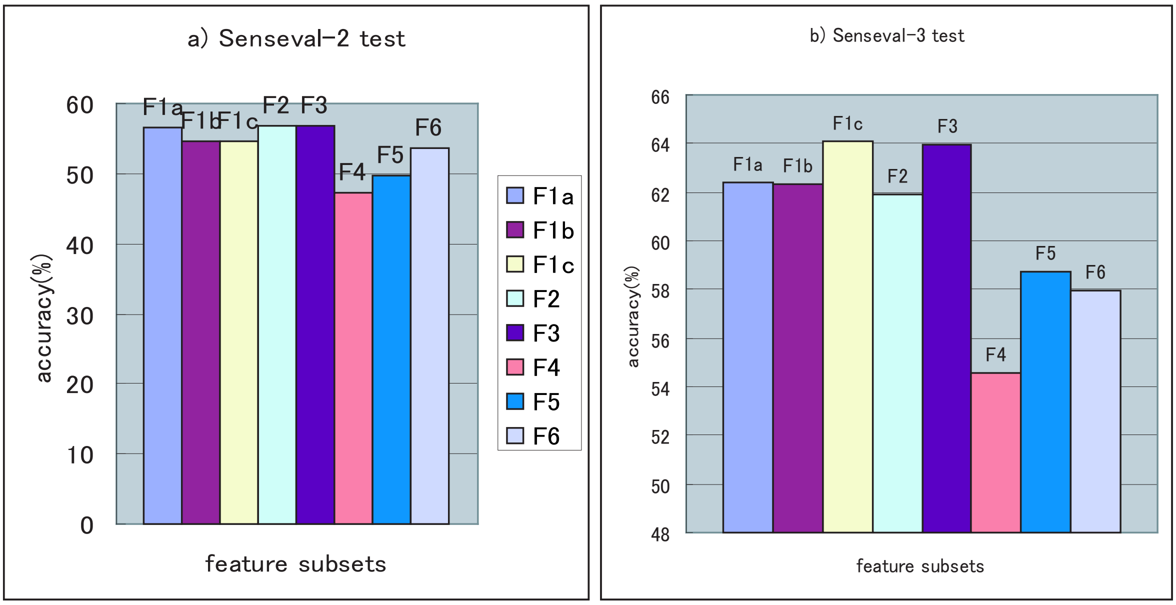


Figure 3.1: An evaluation of feature subsets on Senseval-2 and Senseval-3

In this section we have shown various kinds of considered knowledge. Note that, for topical context, we simultaneously consider three window sizes: (-5,+5) for small context, (-10,+10) for medium context, (-100,+100) for large context (in previous studies, only the large context is utilized). Thus, these five kinds of knowledge as presented above give us eight corresponding subsets of features. Next section will present a method to determine which in these subsets is useful in the overall set of features.

Fig. 3.1 shows the corresponding accuracies of these eight feature subsets when testing on Senseval-2 and Senseval-3. The result shows that bag-of-words, collocations of words, and ordered words give better accuracies in comparison with the remaining feature subsets including collocation of POSs, ordered POSs, and syntactic features.

3.2 Feature Selection

The task is to determine the final set of features. A natural solution is to combine the whole features from the 8 feature subsets, $F_1^a, F_1^b, F_1^c, F_2, F_3, F_4, F_5, F_6$ into a set and consider this set as the final set. However, there may be some noised or redundant features, so it is necessary to find a subset of M relevant features that describes the dataset as well as, or event better than, the original N features do. In our opinion, feature selection for WSD should be treated in two levels.

– In the first level, we deal with the selection on feature subsets to find out which feature subsets are useful for WSD (it is equivalent to the problem of determining which knowledge sources are useful). We consider this problem because when simultaneously consider these knowledge sources for the objective of WSD, there may be some knowledge sources which are redundant or event decrease the accuracy of the WSD task.

– Feature selection of the second level is dealt with individual features. After we know the useful feature subsets at the first level of feature selection, the whole features in these subsets are integrated into a total set of features. And then, we can apply an appropriate method of feature selection on these individual features to check whether there are some redundant ones among them.

Feature selection has long been the focus of researchers of many fields such as pattern recognition, statistics, machine learning. Many methods have been proposed. In general, they can be classified into two categories: filter approach and wrapper approach. In the filter approach [Kira & Rendell (1992)], the feature selector is independent of any learning algorithm and serves as a filter to sieve the irrelevant and/or redundant features based on some measures such as information gain, etc. In the wrapper approach, the feature selector works as a wrapper around a learning algorithm relying on which the relevant features are determined [John *et al.* (1994)]. Regarding two levels of the task of feature selection as presented above, we see that the filter approach is appropriate for the first level and the wrapper approach is appropriate for the second level. Below, we present our method to solve the problem of feature selection for WSD.

3.2.1 Selection of Knowledge Sources

Each knowledge source, in respect to a context, is presented by a subset of features, which can be considered as a representation of the context. Therefore, selecting useful knowledge sources to build the whole set of features means to find the best combination between corresponding feature subsets.

This section presents an extension of one of our papers, [Le & Shimazu (2004)], in which 5 feature subsets were designed and a selection on them was done by using Forward Sequential Selection (FSS) algorithm [Domingos (1997)]. In this thesis, more feature subsets, 8 sets, are considered. In addition, for feature selection we will propose a method which combines two algorithms, forward sequential selection and backward sequential selection (BSS). In the FSS algorithm, we first select the best feature subset according to results tested on all the subsets. next step, we find a subset from the remaining subsets such that the combination between it and the selected subset give the best accuracy in comparison with other combinations. This process will be repeated until all subsets are selected or until the new combinations does not improve the accuracy. In the BSS algorithm, the starting set includes all subsets, and then one subset from the selected

Algorithm 1 – Forward Sequential Selection

```
input: $\mathbf{F} = \{F_1, \ldots, F_n\}$ – a pool of features (or feature subsets)
output: $\mathbf{F}^*$ – the new set of features
$\mathbf{F}^* = \emptyset$
repeat
    $k = \arg\max_{i, F_i \in \mathbf{F}} Eval(\{\mathbf{F}^* \cup F_i\})$
    if $Eval(\mathbf{F}^* \cup F_k) > Eval(\mathbf{F}^*)$ then
        $\mathbf{F}^* = \mathbf{F}^* \cup F_k$
        $\mathbf{F} = \mathbf{F} \setminus F_k$
    else
        $\mathbf{F} = \emptyset$
    end if
until $\mathbf{F} = \emptyset$
```

where $Eval(.)$ is an evaluation procedure which returns the accuracy tested on the predefined training and testing datasets and based on the feature set - the parameter of the procedure. Note that this procedure requires a supervised learning algorithm.

subsets will be removed if the remaining set give the best accuracy. This process will be stopped if we can not improve the accuracy. Particularly, these two algorithms are presented in the Algorithm 1 and Algorithm 2, respectively.

In [Mihalcea (2002)], the author also used the FSS algorithm to carry out a feature selection on some types of features. For a comparison, our proposed method is a combination of FSS and BSS algorithms, thus it can avoid the local optimization obtained by only using FSS. Moreover, the study in [Mihalcea (2002)] worked on the feature types which is too complexity and rarely used in other studies, so this result has not been re-used in WSD community. In contrast, we carry out on the types of features which have been wildly used in previous studies. Our goal is to propose a method to explain for the use of feature selection as in previous studies, or to find out a better selection based on these types of features.

Algorithm 2 – Backward Sequential Selection

```
input: $\mathbf{F} = \{F_1, \ldots, F_n\}$ – a pool of features (or feature subsets)
output: $\mathbf{F}^*$ – the new set of features
$\mathbf{F}^* = \mathbf{F}$
repeat
    $k = \arg\max_{i, F_i \in F^*} Eval(\{\mathbf{F}^* \setminus F_i\})$
    if $Eval(\mathbf{F}^* \setminus F_k) > Eval(\mathbf{F}^*)$ then
        $\mathbf{F}^* = \mathbf{F}^* \setminus F_k$
    end if
until $|\mathbf{F}^*| = 1$ or $\mathbf{F}^*$ not changed
```

where $Eval(.)$ has the same definition as in the algorithm 1.

.

Table 3.1: Results of applying Forward Sequence Selection algorithm

round	selected features	{features/accuracy(%)}
1	{}	F_1^a/73.41 F_1^b/**74.67** F_1^c/72.5 F_2/71.75 F_3/72.42 F_4/64.42 F_5/63.0 F_6/63.33
2	$\{F_1^b\}$	F_1^a/74.58 F_1^c/74.42 F_2/79.17 F_3/**79.75** F_4/74.0 F_5/75.67 F_6/77.17
3	$\{F_1^b, F_3\}$	F_1^a/80.08 F_1^c/8.0 F_2/79.5 F_4/76.25 F_5/78.5 F_6/**80.25**
4	$\{F_1^b, F_3, F_6\}$	F_1^a/**81.17** F_1^c/80.05 F_2/79.75 F_4/76.83 F_5/79.25
5	$\{F_1^b, F_3, F_6, F_1^a\}$	F_1^c/81.0 F_2/80.5 F_4/79.25 F_5/80.5

Table 3.2: Results of applying Backward Sequence Selection algorithm

round	selected features	{features/accuracy(%)}
1	{}	$\{F_1^a, F_1^b, F_1^c, F_2, F_3, F_4, F_5, F_6\}$/81.5
2	$\{F_1^a, F_1^b, F_1^c, F_2, F_3, F_4, F_5, F_6,\}$	$\overline{F_1^a}$/80.83 $\overline{F_1^b}$/80.58 $\overline{F_1^c}$/78.58 $\overline{F_2}$/81.92 $\overline{F_3}$/81.0 $\overline{F_4}$/**83.33** $\overline{F_5}$/81.83 $\overline{F_6}$/81.17
3	$\{F_1^a, F_1^b, F_1^c, F_2, F_3, F_5, F_6,\}$	$\overline{F_1^a}$/83.0 $\overline{F_1^b}$/82.5 $\overline{F_1^c}$/81.25 $\overline{F_2}$/82.5 $\overline{F_3}$/82.67 $\overline{F_5}$/82.08 $\overline{F_6}$/83.0

Experiment

In the experiment, we choose Naive Bayes algorithm for the supervised learning algorithm in the procedure $Eval(.)$ in the FSS and BSS procedures.

We use datasets of the four words, *interest*, *hard*, and *serve* as the development datasets. For each dataset corresponding to each word, we randomly select 200 examples for training data and 100 examples for test data. These selected sizes are chosen for being compatible to the sizes of Senseval-2 and Senseval-3.

For these development data, we implement the Algorithm 1 and Algorithm 2 with the 8 feature sets, $F_1^a, F_1^b, F_1^c, F_2, F_3, F_4, F_5, F_6$. The obtained results are the average results of testing on the four words, and they are shown in Table. 3.1 and Table. 3.2.

Table 3.1 shows the result of applying the FSS algorithm. The best result is obtained with the pool of feature subsets, $\{F_1^b, F_3, F_6\}$, which is corresponding to the accuracy 81.17%. Table 2 shows the result of applying the BSS algorithm. The best result is obtained with the pool of feature subsets, $\{F_1^a, F_1^b, F_1^c, F_2, F_3, F_5, F_6,\}$, corresponding with the accuracy 83.3%. In this case, BSS algorithm gives a better result, and therefore this result is chosen as the final selection of the pool of context representations (feature subsets). It is worth to emphasize that most (7 among 8) feature subsets are selected, exception of the feature set F_4 – the collocations of POSs. This result also shows the effectiveness of simultaneously consideration of topical context representation on all three kinds: small, medium, and large sizes.

This result is used for testing on Senseval-2 and Senseval-3 and the results are showed in Table 3.3. In this test we use more others supervised learning than NB, including Sup-

Table 3.3: Test on Senseval-2 and Senseval-3

	selected subsets			all subsets			as previous studies		
	NB	SVM	MEM	NB	SVM	MEM	NB	SVM	MEM
Senseval-2	64.05	63.72	**64.79**	63.40	63.22	63.98	62.45	62.25	62.68
Senseval-3	**71.96**	70.87	71.91	70.97	70.06	**71.98**	70.06	69.88	70.16
Average	68.005	67.295	**68.35**	67.185	66.64	67.98	66.25	66.06	66.42

port Vector Machines (SVMs) and Maximum Entropy Model (MEMs). For a comparison, we carry out a test with all features from the 8 subsets, $\{F_1^a, F_1^b, F_1^c, F_2, F_3, F_4, F_5, F_6, \}$, and also carry out a test in which the feature set is selected similarly to the feature set in [Lee & Ng(2002)] (i.e. the set of $\{F_1^c, F_2, F_3, F_5, F_6, \}$).

From the results in Table 3.3, we can extract the following conclusions:

- In most cases, the selected combination of subsets give better result in comparison with using all subsets, except the case of using MEM for Senseval-3, but with a slight difference. This result is also better the result obtained by using the feature set as presented in previous studies such as in [Lee & Ng(2002)].

- This result also suggests to use NB and MEM for supervised WSD because their results are nearly the same and better in comparison with SVM's result.

3.2.2 Selection of Individual Features

Suppose that we have already had the whole set of features obtained from selected knowledge sources. As the objective of feature selection we will try to reduce as many as possible the number of features while keeping the accuracy of the classification (i.e. determining word senses). This task should satisfy that it reduce the native space of features without sacrificing the classification (sense disambiguation) accuracy. The wrapper approach is not appropriate for a very large set of features due to it costs a lot of computation time. As discussed before, we choose the filter approach. The filter approach selects features based on using some measures to evaluate features. [Yang & Pedersen (1997)] did a comparative study on feature selection for text categorization which followed the filter approach with various kinds of feature measuring including document frequency, information gain, mutual information, a χ-test, and term strength. Concerning related works in WSD, [Lee & Ng(2002), Mihalcea (2004), Pham *et al.* (2005)] just chose the features which have their frequency being equal or greater than 3. Among them, [Lee & Ng(2002)] uses both cases: all features and only features with their frequencies is equal or greater than 3, and their result shows that using all features gives better results. In this work, we will investigate two well-known measures of features including frequency and information-gain.

Feature Measure using Frequency

We let the frequency threshold runs from 1 to 4 and test on Senseval-2 and Senseval-3 using NB classifier. With each frequency τ, the selected features will include all features having their frequencies being equal or greater than τ. The results are shown in Fig. 3.2, in which the part a) is the result of testing on Senseval-2 and the part b) is the the result of testing on Senseval-3. These results show that when the frequency threshold increases the accuracy quickly decreases, it means that reducing features of low frequency will decrease accuracy of WSD.

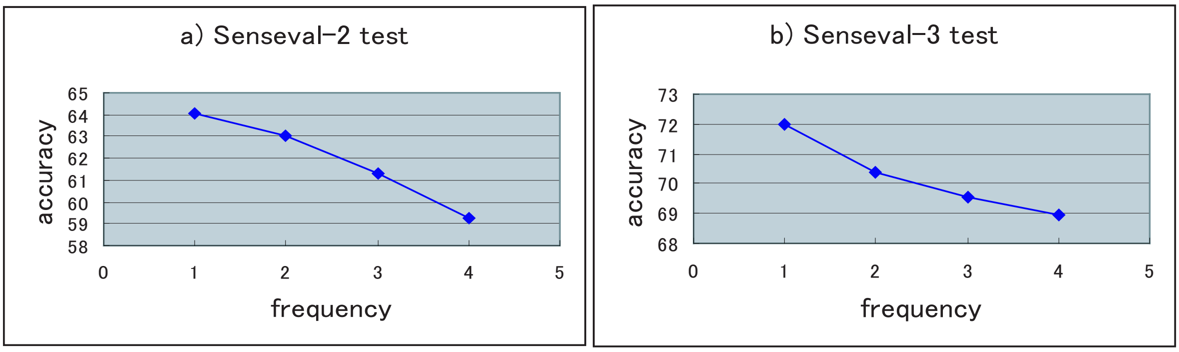


Figure 3.2: A test on Senseval with feature selection based on frequency

Feature Measure using Information Gain

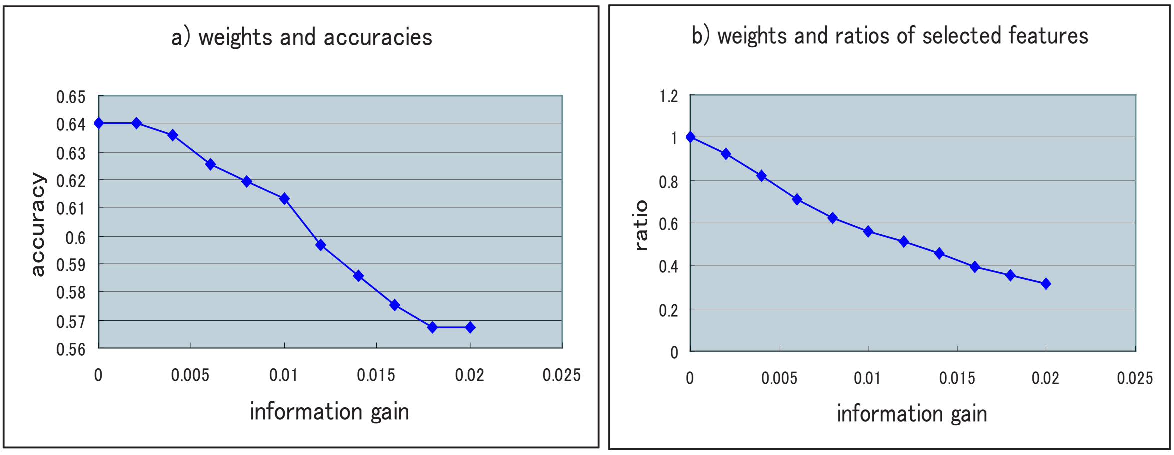


Figure 3.3: A test on Senseval-2 with feature selection based on information gain

Information gain is frequently employed as a term-goodness criterion in the field of machine learning. It measures the number of bits of information obtained for classification. Let $S = \{s_1, \ldots, s_m\}$ denotes the set of classes (or senses). We follow the definition of information gain of a term t as presented in [Yang & Pedersen (1997)], as follows:

$$IG(t) = -\sum_{i=1}^{m} P_r(s_i) log P_r(c_i) + P_r(t) \sum_{i=1}^{m} P_r(c_i|t) log P_r(c_i|t) + P_r(\bar{t}) \sum_{i=1}^{m} P_r(c_i|\bar{t}) log P_r(c_i|\bar{t})$$

where $\bar{t}$ denote the complementary of t, i.e. $P_r(\bar{t}) = 1 - P_r(t)$ and $P_r(c_i|\bar{t})$ defines the conditional probability of c_i without co-occurrence of the term t.

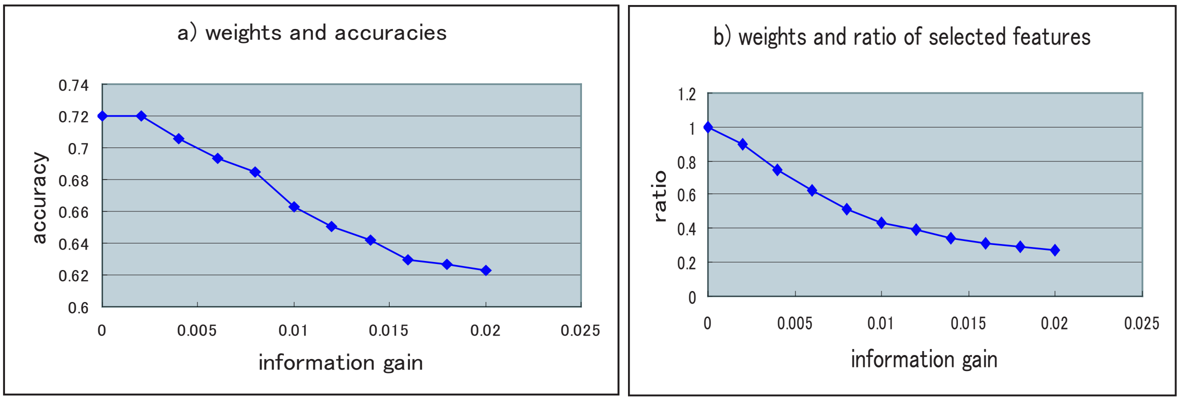


Figure 3.4: A test on Senseval-3 with feature selection based on information gain

Table 3.4: A comparison on Senseval-2 and Senseval-3

	NB	SVM	MEM	The best system in Senseval contests	Baseline
Senseval-2	64.05	63.72	64.79	64.2	48.35
Senseval-3	71.96	70.87	71.91	72.9	55.17

Fig. 3.3 and Fig. 3.4 show our experiments in which a threshold α for information gain is used for determining the features will be used. For each α we select the features which have their information gain is equal or greater than α. In the experiment, we let α receives values in $\{0, 0.002, 0.004, 0.006, 0.008, 0.01, 0.012, 0.014, 0.016, 0.018, 0.02\}$. Fig. 3.3 shows the result when testing on Senseval-2, in which the part a) shows the obtained accuracies corresponding to the values of α, and the part b) shows the ratio of number of selected features on all original features (note that the part b) provides us an image about the reduction of features). The same experiment for Senseval-3 is presented in Fig. 3.4. From these results we can see that when we increase the value of α (i.e. the number of selected features is decreased) the obtained accuracies decrease.

In summary, we see that when we reduce the number of features which are based on frequency or information gain, the accuracy of word sense disambiguation is decreased.

3.3 Summary

In this chapter we have presented our work on context representation and feature selection for word sense disambiguation.

The first task, context representation, is the problem of determining which kinds of knowledge extracted from context is useful for the task of word sense disambiguation, or is the problem of selecting useful knowledge sources. We have investigated various knowledge sources used in the recent WSD studies such as [Lee & Ng(2002), Ando(2006), Montoyo *et al.* (2005)]. In addition, different to previous studies, we have simultaneously used the three representations of bag-of-words based on three window sizes including small size, medium size, and large size. All these kinds of knowledge are represented by subsets of features (which can be considered as different context representations), and then will be used in a feature selection method to find the best combination of them.

The second task is the problem of feature selection. We divide this task into two subtasks, including the selection of knowledge sources (represented by feature subsets) and the selection of individual features. For the first subtask, we proposed a method that uses concurrently the forward sequential selection and the backward sequential selection algorithms to get the best combination of the feature subsets obtained above. As the result, we have obtained the final set of features, which includes the whole features from the subsets $\{F_1^a, F_1^b, F_1^c, F_2, F_3, F_5, F_6, \}$. For the second subtask, we used the filter approach and based on frequency-based and based-information-gain measures. Experimental results show that reducing some features based on these two measures will decrease accuracy of WSD. In other word, the experiment suggests us to use all features from the selected knowledge sources.

Table. 3.4 shows a test using NB, MEM, and SVM algorithms with the obtained feature set. The obtained results can be comparable with the best systems on Senseval-2 and Senseval-3 contests.

Chapter 4

Classifier Combination

During the last decade, many supervised machine learning algorithms have been used for the WSD task, including Naïve Bayesian (NB), decision trees, an exemplar-based, support vector machine, maximum entropy, etc. As observed in studies of machine learning systems, although one of the available learning systems could be chosen to achieve the best performance for a given pattern recognition problem, the set of patterns misclassified by the different classification systems would not necessarily overlap. This means that different classifiers may potentially offer complementary information about patterns to be classified. This observation highly motivated the recent interest in combining classifiers.

This chapter presents our study in applying multi-classifier combination for word sense disambiguation. First, we introduce classifier combination architecture, related work, the motivation, and our proposal. Next, we present a general framework of combination which can be used for both types of individual classifiers: based on different feature sets, and based on different supervised machine learning algorithms. Then, we formulate the common combination rules, such as majority voting, the so-called Naive-Bayesian-based rule (or shortly NB rule), etc., in this framework. Two new approaches of classifier combination for WSD, which are based on Dempster-Shafer theory of evidence and based on OWA operators, are introduced in next sections. After that, we present two proposed schemes for second-layer combination strategies. Finally, we present different experiments on various combination strategies as mentioned in the before sections.

4.1 Introduction

4.1.1 Architecture of Multi-Classifier Combination

Classifier combination has been studied intensively in the last decade, and has been shown to be successful in improving performance on diverse applications [Brill & Wu (1998), Pedersen (2000), Kilgarriff & Rosenzweig (2000), Klein *et al.* (2002)]. The intuition behind classifier combination is that individual classifiers have different strengths and perform well on different subtypes of test data. According to [Kittler *et al.* (1998)], there are basically two classifier combination scenarios. In the first scenario, all classifiers use the same representation of the input pattern. In the second scenario, each classifier uses its own representation of the input pattern. Moreover, generation of different classifiers can be based on the different training datasets. Fig. 4.1 intuitively present the architecture of multiple classifier combination.

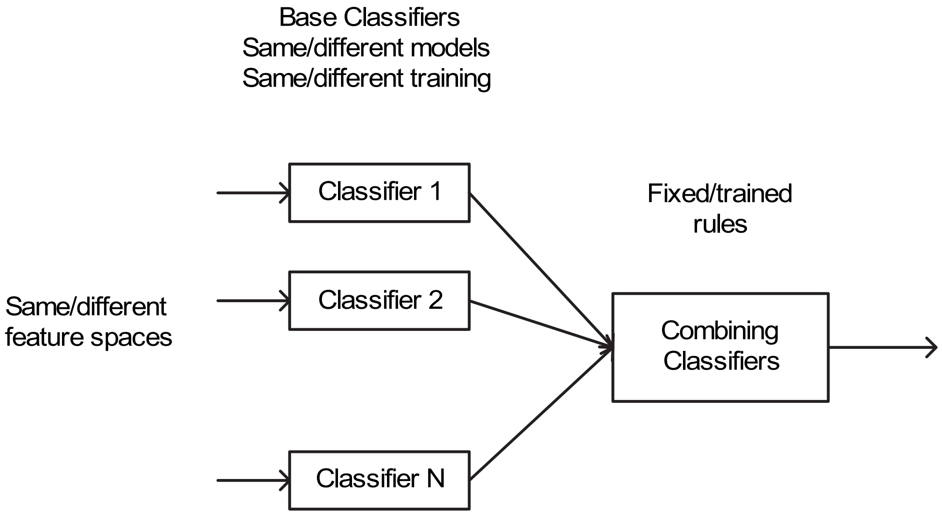


Figure 4.1: Architecture of multiple classifier combination

From this figure, we can see that the base classifiers (also called individual classifiers) can be created based on the different feature spaces, different training datasets, or different models (machine learning algorithms). Note that by combining these different types, we can also create other set of individual classifiers, however this method is rarely used in previous studies. Fig. 4.1 also shows the general process of applying classifier combination strategies for a problem such as classification. That is, firstly the set of individual classifiers are built and then they are used for detecting test examples. Outputs of these individual classifiers are then combined using fixed combination rules or training combination rules to generate consensus decisions. For a more detail, we investigate some approaches in classifier combination as below.

In the **bagging** approach, the training set for each individual classifier is created by randomly drawing training examples with replacement from the initial training set. In **boosting**, the errors made by a classifier learned from a training set are used to construct a new training set in which the misclassified examples get higher weight. By sequentially performing this operation, an ensemble is constructed. Another way to create multiple classifiers is based on multiple feature sets on the same training dataset. Methods of combining the outputs of component classifiers in an ensemble include **simple voting**, wherein each component classifier gets an equal vote, and **weighted voting**, in which each component classifier's vote is weighted by its accuracy. An interesting approach to combination is **stacking**, in which we use all individual classifiers to detect labels for each training example, and the sequence of these labels are then used to represent for this example. Consequently, the training dataset is replaced by the set of corresponding label sequences and they are used to train the final classifier. For testing step, the input of each test example for this final classifier is the outputs of using the individual classifier for this test example. The scheme of staking method is sketched in Fig. 4.2.

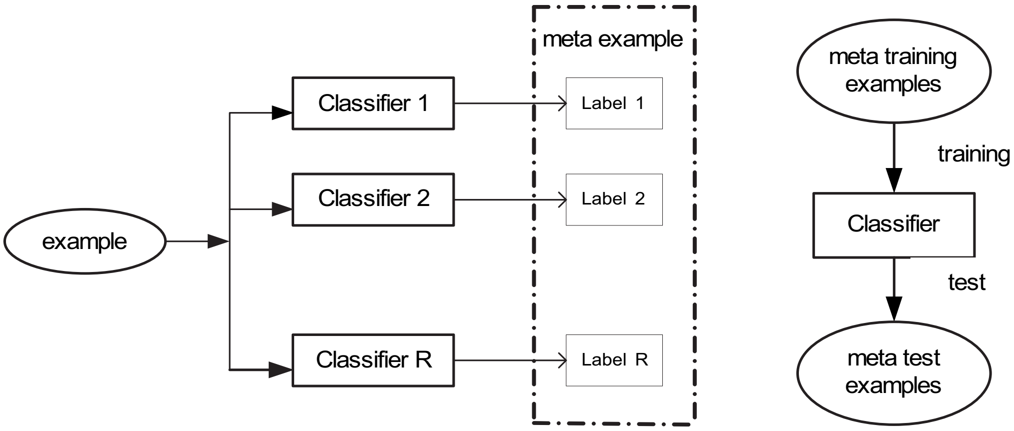


Figure 4.2: Stacking method of classifier combination

4.1.2 Related Work and Proposal

In the WSD literature, the first empirical study of combining classifiers was presented in [Kilgarriff & Rosenzweig (2000)], in which the authors combined the output of the participating SENSEVAL1 systems via simple voting. [Pedersen (2000)] built an ensemble of Naive Bayesian classifiers, each of which is based on lexical features that represent co-occurring words in varying sized windows of context. [Klein *et al.* (2002)] use a stacking type of combination techniques. First, individual classifiers were constructed based on different training datasets and learning methods, and then they were ranked according to results obtained from testing on held-out data. In the next step, majority voting, weighted voting, and maximum entropy were used as combination strategies. [Hoste *et al.* (2002)] used word experts consisting of four memory-based learners trained on different context. Output of the word experts is based on majority voting or weighted voting. [Wang & Matsumoto (2004)] presented a kind of stacking; individual classifiers were built using NB with varying sized windows of context that are similar to Pedersen's approach [Pedersen (2000)], and then used K-nearest neighbors as the meta learning method. Several combination approaches in WSD, most of which used majority voting or weighted voting on the output of individual classifiers, were based on different sets of features or different learning methods. Some of them proposed different approaches such as using maximum entropy [Klein *et al.* (2002)], or a stacking method [Wang & Matsumoto (2004)] as combination strategies.

One well-known study about using multi-classifier combination for word sense disambiguation is [Florian & Yarowsky (2002)]. In this paper, the authors used six different classifiers, which as generated by using six machine learning methods, as components of their combination. Combination strategies introduced in this paper are based on the weighted mixture model. Among them, some strategies combine the posterior sense probability distributions in which the weights are yielded based on Expectation-Maximization (EM) algorithm, Performance-Based (PB) algorithm (see [Florian & Yarowsky (2002)] for detail). One more combination strategy are presented in which the synthetic sense probability distribution are computed based on sense ranks computed from individual classifiers. This strategy of combination is called rank-based combination. Finally, they use a meta combination which gives the best accuracy. This meta-combination is the av-

Table 4.1: Approaches in previous WSD studies on multi-classifier combination

Paper	**Method**	**Individual classifiers based on ...**
[Pedersen (2000)]	majority voting	bag-of-words different windows
[Klein *et al.* (2002)]	majority voting weighted voting maximum entropy & stacking	different ML models ML models
[Florian & Yarowsky (2002)]	mixture models rank-based average-based meta	different ML models
[Wang & Matsumoto (2004)]	kNN-based stacking	bag-of-words different windows

erage of results of the three combinations which are EM-based, PB-based, and rank-based algorithms.

In summary, the Table. 4.1 shows some popular WSD studies with corresponding methods on classifier combination.

Previous studies have shown that multi-classifier combination methods can improve performances of supervised WSD systems. However, these studies just focus on several combination strategies, such as majority voting, weighted voting, simple mixture models, and a staking methods using maximum entropy model [Klein *et al.* (2002)] ore kNN [Wang & Matsumoto (2004)]. There are still other combination strategies which have not been investigated for WSD yet, even some popular strategies such as product rule, max rule, min rule, the so-called Naive Bayesian rule, etc., as introduced in [Kittler *et al.* (1998)]. Furthermore, approaches in previous studies seem to be lack of describing a general combination framework with a corresponding theoretical basis, which should be the fundamentals for applying the combination rules used. These limitations motivate us to apply multi-classifier combination for WSD with a comprehensive collection of combination rules in certain theoretical basis.

Particularly, we will present a general framework of combination for WSD with basically combination rules including NB-based rule, majority voting, weighted voting, and average rule. Then, by considering WSD problem in classifier combination as the problem of weighted combination of evidence for decision making, we formulate a general rule of classifier combination based on DS theory of evidence [Shafer (1976)], adopting a probabilistic interpretation of weights. This interpretation of weights seems to be appropriate when defining weights in terms of the accuracy of individual classifiers. On the other hand, by considering individual classifiers as information inspired by a semantics or syntactical criterion for the purpose of word sense identification, we can apply OWA operators for aggregating multi-criteria to form an overall decision function considered as the fuzzy majority based voting strategy. It should be worth mentioning that the use of OWA operators in classifier combination has been studied, for example, in [Kuncheva 2001]. In this work, however, we use OWA operators for classifier fusion in their semantic relation to linguistic quantifiers so that we could provide a framework for combining classifiers, which also yields several commonly used decision rules but without some strong assumptions made in the work by [Kittler *et al.* (1998)]. Next, we also present some new variants of meta-combination and stacking methods. Finally, experiments and conclusions are presented.

4.2 Common Combination Strategies

4.2.1 General Framework of Multi-Classifier Combination

Suppose w is the polysemous word to be disambiguated. Let $D = \{D_1, \ldots, D_R\}$ be a set of classifiers, and let $S = \{c_1, \ldots, c_M\}$ be a set of potential labels (senses) of w. Alternatively, we may define the classifier output to be a M-dimensional vector

$$D_i(w) = [d_{i,1}(w), \ldots, d_{i,M}(w)] \tag{4.1}$$

where $d_{i,j}(w)$ is the degree of "support" given by classifier D_i to the hypothesis that w comes from class c_j. Most often $d_{i,j}(w)$ is an estimation of the posterior probability $P(c_i|w)$. In fact, the detailed interpretation of $d_{i,j}(w)$ beyond a "degree support" is not important for the operation for any of the combination methods studies here. It is convenient to organize the outputs of all R classifiers in a decision matrix as follows.

$$DP(w) = \begin{bmatrix} d_{1,1}(w) & \ldots & d_{1,j}(w) & \ldots & d_{1,M}(w) \\ \ldots & & & & \\ d_{i,1}(w) & \ldots & d_{i,j}(w) & \ldots & d_{i,M}(w) \\ \ldots & & & & \\ d_{R,1}(w) & \ldots & d_{R,j}(w) & \ldots & d_{R,M}(w) \end{bmatrix} \tag{4.2}$$

Thus, the output of classifier D_i is the i-th row of the decision matrix, and the support for class c_j is the jth column. *Combining classifiers* means to find a class label based on the R classifiers outputs. We look for a vector with M final degrees of support for the classes, denoted by

$$D(w) = [\mu_1(w), \ldots, \mu_M(w)] \tag{4.3}$$

where $\mu_j(w)$ is the overall support degree obtained by combining R support degrees $\{d_{1,j}(w), \ldots, d_{R,j}(w)\}$ from outputs of the R individual classifiers, under a combination operator $\oplus$, as presented in the formula (4.4) below.

$$\mu_j(w) = \oplus(d_{1,j}(w), \ldots, d_{R,j}(w)), \ for\ j = 1, \ldots, M \tag{4.4}$$

If a single class label of w is needed, we use the maximum membership rule: Assign w to class c_k iff

$$\mu_k(w) \geq \mu_t(w), \forall t = 1, \ldots, M. \tag{4.5}$$

It is interesting that if we assume that each individual classifier D_i is based on a knowledge source, namely $\mathbf{f}_i$, and then when detecting senses for w, the classifier D_i outputs a probability distribution over the label set S, denote by $\{P(c_j|\mathbf{f}_i)\}_{j=1}^{M}$. In other word, to distinguish the individual classifiers we assume each classifier D_i is based on a knowledge source $\mathbf{f}_i$, and consequently we can represent $d_{i,j}(w)$ by the posterior probability, $P(c_j|\mathbf{f}_i)$, or by following equation:

$$d_{i,j}(w) = P(c_j|\mathbf{f}_i) \tag{4.6}$$

Note that this representation is used for all types of generation of different individual classifiers event though it is more appropriate for individual classifiers based on different feature spaces than different machine learning algorithms.

4.2.2 Naive Bayesian and Product Rules

Under a mutually exclusive assumption of a set $\mathbf{F} = \mathbf{f}_i$ $(i = 1, \dots, R)$, the Bayesian theory suggests that the word w should be assigned to class c_k provided the a posteriori probability of that class is maximum, namely

$$k = \arg\max_j P(c_j|\mathbf{f}_1, \dots, \mathbf{f}_R) \tag{4.7}$$

That is, in order to utilize all the available information to reach a decision, it is essential to consider all the representations of the target simultaneously.

The decision rule (4.7) can be rewritten using Bayes theorem as follows:

$$k = \arg\max_j \frac{P(\mathbf{f}_1, \dots, \mathbf{f}_R|c_j)P(c_j)}{P(\mathbf{f}_1, \dots, \mathbf{f}_R)}$$

Because the value of $P(\mathbf{f}_1, \dots, \mathbf{f}_R)$ is unchanged with variance of c_j, we have

$$k = \arg\max_j P(\mathbf{f}_1, \dots, \mathbf{f}_R|c_j)P(c_j) \tag{4.8}$$

As we see, $P(\mathbf{f}_1, \dots, \mathbf{f}_R|c_j)$ represents the joint probability distribution of the knowledge sources corresponding to the individual classifiers. Assume that these knowledge sources are conditionally independent, so that the decision rule (4.8) can be rewritten as follows:

$$k = \arg\max_j P(c_j)\prod_{i=1}^{R} P(\mathbf{f}_i|c_j) \tag{4.9}$$

According to Bayes rule, we have:

$$P(\mathbf{f}_i|c_j) = \frac{P(c_j|\mathbf{f}_i)P(\mathbf{f}_i)}{P(c_j)} \tag{4.10}$$

Naive Bayesian Rule

Substituting (4.10) into (4.9), we obtain the Naive Bayesian (NB) Rule:

$$k = \arg\max_j [P(c_j)]^{-(R-1)} \prod_{i=1}^{R} P(c_j|\mathbf{f}_i) \tag{4.11}$$

It is worth to emphasize that we can consider NB rule as the NB classifier built on the combination of all feature subsets $\mathbf{f}_1, \dots, \mathbf{f}_R$.

Product Rule

Like [Kittler *et al.* (1998)], if we assume that all prior probabilities $P(c_j)$ (j=1,...,M) are of equal value, and then we obtain the equation, as follows.

$$k = \arg\max_j \prod_{i=1}^{R} P(c_j|\mathbf{f}_i) \tag{4.12}$$

The decision rule (4.12) quantifies the likelihood of a hypothesis by combining the a posteriori probabilities generated by the individual classifiers by means of a *product rule*.

4.2.3 Median/Average Rule

Let us return to the decision matrix (4.2), each classifier D_i supports a degree $d_{i,j}(w)$ for the class c_j. According to [Perrone & Cooper (1993)], if the errors made by R classifiers D_i, $i = 1, \ldots, R$, are uncorrelated and unbiased, then these R classifiers can be combined into a classifier that supports the class c_j with the degree

$$\mu_j(w) = \left[\frac{1}{R}\sum_{i=1}^{R} d_{i,j}(w)\right] \tag{4.13}$$

According to (4.6), $\mathbf{f}_i$ is the knowledge source of w with respect to the classifier D_i, equation (4.13) then becomes

$$\mu_j(w) = \left[\frac{1}{R}\sum_{i=1}^{R} P(c_j|\mathbf{f}_i)\right] \tag{4.14}$$

Therefore, the class (sense) c_k is chosen as the best class for the target word under the *median rule* (or *average rule*) as follows

$$k = \arg\max_j \left[\frac{1}{R}\sum_{i=1}^{R} P(c_j|\mathbf{f}_i)\right] \tag{4.15}$$

4.2.4 Majority Voting and Weighted Voting

Majority voting follows a simple rule as: it will vote for the class which is chosen by maximum number of individual classifiers. This can be done by hardening the a posteriori probabilities $P(c_k|\mathbf{f}_i)$ in terms of functions Δ_{ki} defined as follows

$$\Delta_{ki} = \begin{cases} 1, & \text{if } P(c_k|\mathbf{f}_i) = \max_j P(c_j|\mathbf{f}_i) \\ 0, & \text{otherwise} \end{cases}$$

then the right class (sense) c_j is determined as follows:

$$j = \arg\max_k \sum_{i=1}^{R} \Delta_{ki} \tag{4.16}$$

Weighted voting can be considered as a "soft" majority voting, in which instead of hardening the a posteriori probabilities $P(c_k|\mathbf{f}_i)$ into 1 in the case the classifier i^{th} chooses

c_k, weighted voting still keep this value. With this principle, the functions Δ_{ki} defined as follows

$$\Delta_{ki} = \begin{cases} P(c_k|\mathbf{f}_i), & \text{if } P(c_k|\mathbf{f}_i) = \max_j P(c_j|\mathbf{f}_i) \\ 0, & \text{otherwise} \end{cases}$$

and then, the decision rule of weighted voting is also the same as (4.16). It is worth to note that weighted voting is specially more effective than majority voting in the case there are more than one class with the same priority.

4.3 Combination Based on Dempster-Shafer Theory of Evidence

In this section we discuss a framework for weighted combination of classifiers for WSD, in which each individual classifiers is considered as based on different information sources. Moreover, each of these information sources does not by itself provide 100% certainty as a whole piece of evidence for identifying the sense of the target. Under such an observation, we have interpreted the framework of classifier combination in terms of Dempster-Shafer theory of evidence [Shafer (1976)], and then formulated a general rule of classifier combination from which several interestingly classifier combination schemes are derived. In [Al-Ani & Deriche 2002], Al-Ani and Deriche have proposed a new technique for combining classifiers using DS theory, in which different classifiers correspond to different feature sets. In their approach, the distance between the output classification vector provided by each single classifier and a reference vector are used to estimate Basic Probability Assignments (BPAs). These BPAs are then combined making use of Dempster's rule of combination to obtain a new output vector that represents the combined confidence in each class label. Different from their approach, we directly use the output classification vectors of individual classifiers to define the corresponding BPAs, making use of the discount operation in DS theory and then combine the resulted BPAs to obtain the final BPA for making the decision of classification.

4.3.1 Basic Concepts

In DS theory, a problem domain is represented by a finite set Θ of mutually exclusive and exhaustive hypotheses, called *frame of discernment* [Shafer (1976)]. In the standard probability framework, all elements in Θ are assigned a probability. And when the degree of support for an event is known, the remainder of the support is automatically assigned to the negation of the event. On the other hand, in DS theory mass assignments are carried out for events as they know, and committing support for an event does not necessarily imply that the remaining support is committed to its negation. Formally, a Basic Probability Assignment (BPA, for short) is a function $m : 2^\Theta \to [0, 1]$ verifying

$$m(\emptyset) = 0, \text{ and } \sum_{A \in 2^\Theta} m(A) = 1$$

The quantity $m(A)$ can be interpreted as a measure of the belief that is committed exactly to A, given the available evidence. A subset $A \in 2^\Theta$ with $m(A) > 0$ is called a *focal element* of m. A BPA m is called to be *vacuous* if $m(\Theta) = 1$ and $m(A) = 0$ for all $A \neq \Theta$.

Two evidential functions derived from the basic probability assignment m are the belief function Bel_m and the plausibility function Pl_m, defined as

$$Bel_m(A) = \sum_{\emptyset \neq B \subseteq A} m(B), \text{ and } Pl_m(A) = \sum_{B \cap A \neq \emptyset} m(B)$$

The difference between $m(A)$ and $Bel_m(A)$ is that while $m(A)$ is our belief committed to the subset A excluding any of its proper subsets, $Bel_m(A)$ is our degree of belief in A as well as all of its subsets. Consequently, $Pl_m(A)$ represents the degree to which the evidence fails to refute A. Note that all the three functions are in an one-to-one correspondence with each other.

Two useful operations that play a central role in the manipulation of belief functions are *discounting* and *Dempster's rule of combination* [Shafer (1976)]. The discounting operation is used when a source of information provides a BPA m, but one knows that this source has probability α of reliable. Then one may adopt $(1-\alpha)$ as one's *discount rate*, which results in a new BPA m^α defined by

$$\begin{aligned} m^\alpha(A) &= \alpha m(A), \quad \text{for any } A \subset \Theta && (4.17)\\ m^\alpha(\Theta) &= (1-\alpha) + \alpha m(\Theta) && (4.18) \end{aligned}$$

Consider now two pieces of evidence on the same frame Θ represented by two BPAs m_1 and m_2. Dempster's rule of combination is then used to generate a new BPA, denoted by $(m_1 \oplus m_2)$ (also called the orthogonal sum of m_1 and m_2), defined as follows

$$\begin{aligned} (m_1 \oplus m_2)(\emptyset) &= 0, \\ (m_1 \oplus m_2)(A) &= \tfrac{1}{1-\kappa} \sum_{B \cap C = A} m_1(B) m_2(C) \end{aligned} \qquad (4.19)$$

where

$$\kappa = \sum_{B \cap C = \emptyset} m_1(B) m_2(C) \qquad (4.20)$$

Note that the orthogonal sum combination is only applicable to such two BPAs that verify the condition $\kappa < 1$.

4.3.2 DS Theory Based Combination Scheme

Given a target word w in a context C and $\mathcal{S} = \{c_1, c_2, \ldots, c_M\}$ is the set of its possible senses. Using the vocabulary of DS theory, $\mathcal{S}$ can be called the *frame of discernment* of the problem. As mentioned above, suppose that each individual classifier is based on an information source $\mathbf{f}_i$. Each of these information sources does not by itself provide 100% certainty as a whole piece of evidence for identifying the sense of the target. Formally, we have the available information for making the final decision on the sense of w given as follows

- R probability distributions $P(\cdot|\mathbf{f}_i)$ $(i = 1, \ldots, R)$ on $\mathcal{S}$,

- the weights α_i of the individual information sources $(i = 1, \ldots, R)$[1].

[1] Note that the constraint $\sum_i \alpha_i = 1$ does not need to be imposed.

From the probabilistic point of view, we may straightforwardly think of the combiner as a weighted mixture of individual classifiers defined as

$$P(c_k) = \frac{1}{\sum_i \alpha_i} \sum_{i=1}^{R} \alpha_i P(c_k|\mathbf{f}_i), \text{ for } k = 1, \ldots, M \tag{4.21}$$

Then the target word $\mathbf{w}$ should be naturally assigned to the sense c_j according to the following decision rule

$$j = \arg\max_{k} P(c_k) \tag{4.22}$$

However, by considering the problem as that of weighted combination of evidence for decision making, we now formulate a general rule of combination based on DS theory. To this end, we first adopt a probabilistic interpretation of weights. That is, the weight α_i ($i = 1, \ldots, R$) is interpreted as reliable probability of the i-th classifier. This interpretation of weights seems to be especially appropriate when defining weights in terms of the accuracy of individual classifiers.

Under such an interpretation of weights, the piece of evidence represented by $P(\cdot|\mathbf{f}_i)$ should be discounted at a discount rate of $(1 - \alpha_i)$. This results in a BPA m_i defined by

$$\begin{aligned} m_i(\{c_k\}) &= \alpha_i P(c_k|\mathbf{f}_i) \triangleq p_{i,k}, \text{ for } k = 1, \ldots, M && (4.23)\\ m_i(\mathcal{S}) &= 1 - \alpha_i \triangleq p_{i,\mathcal{S}} && (4.24)\\ m_i(A) &= 0, \forall A \in 2^{\mathcal{S}} \setminus \{\mathcal{S}, \{c_1\}, \ldots, \{c_M\}\} && (4.25) \end{aligned}$$

That is, the discount rate of $(1 - \alpha_i)$ should not be distributed to anything else than $\mathcal{S}$, the whole frame of discernment.

We are now ready to formulate our belief on the decision problem by aggregating all pieces of evidence represented by m_i's in the general form of the following

$$m = \bigoplus_{i=1}^{R} m_i \tag{4.26}$$

where m is a BPA and $\oplus$ is a combination operator in general.

By applying different combination operations for $\oplus$, we may have different aggregation schemes for obtaining the BPA m which models our belief for making the decision on the sense of $\mathbf{w}$. Therefore, we must also deal with the problem of how to make a decision based on m. As m does not in general provide a unique probability distribution on $\mathcal{S}$, but only a set of *compatible probabilities* bounded by the belief function Bel_m and the plausibility function Pl_m. Consequently, individual classes in $\mathcal{S}$ can no longer be ranked according to their probability. Fortunately, based on the *Generalized Insufficient Reason Principle*, we may define a probability function P_m on $\mathcal{S}$ derived from m for the purpose of decision making via the *pignistic transformation* [Smets (1994)]. That is, as in the two-level language of the so-called *transferable belief model* [Smets (1994)], the aggregated BPA m itself representing the belief is entertained based on the available evidence at the *credal level*, and when a decision must be made, the belief at the credal level induces the probability function P_m for decision making.

4.3.3 The Discounting-and-Orthogonal Sum Combination Strategy

As discussed above, we consider each $P(\cdot|\mathbf{f}_i)$ as the belief quantified from the information source $\mathbf{f}_i$ and the weight α_i as a “degree of trust” of $\mathbf{f}_i$ supporting the identification for the sense of w as a whole. As mentioned in [Shafer (1976)], an obvious way to use discounting with Dempster’s rule of combination is to discount all BPAs $P(\cdot|\mathbf{f}_i)$ $(i = 1, \dots, R)$ at corresponding rates $(1 - \alpha_i)$ $(i = 1, \dots, R)$ before combining them.

Thus, Dempster’s rule of combination now allows us to combine BPAs m_i $(i = 1, \dots, R)$ under the independent assumption of information sources for generating the BPA m, i.e. $\oplus$ in (4.26) is the orthogonal sum operation.

Note that, by definition, focal elements of each m_i are either singleton sets or the whole set $\mathcal{S}$. It is easy to see that m also verifies this property if applicable. Interestingly, the commutative and associative properties of the orthogonal sum operation with respect to a combinable collection of BPAs m_i $(i = 1, \dots, M)$ and the mentioned property essentially form the basis for developing a recursive algorithm for calculation of the BPA m [Yang & Xu 2002]. This can be done as follows.

Let $I(i) = \{1, \dots, i\}$ be the subset consisting of first i indexes of the set $\{1, \dots, R\}$. Assume that $m_{I(i)}$ is the result of combining the first i BPAs m_j, for $j = 1, \dots, i$. Let us denote

$$p_{I(i),k} \triangleq m_{I(i)}(\{c_k\}), \text{ for } k = 1, \dots, M \tag{4.27}$$

$$p_{I(i),\mathcal{S}} \triangleq m_{I(i)}(\mathcal{S}) \tag{4.28}$$

With these notations and (4.23)–(4.24), the key step in the combination algorithm is to inductively calculate $p_{I(i+1),k}$ $(k = 1, \dots, M)$ and $p_{I(i+1),\mathcal{S}}$ as follows

$$p_{I(i+1),k} = \frac{1}{\kappa_{I(i+1)}}[p_{I(i),k}p_{i+1,k} + p_{I(i),k}p_{i+1,\mathcal{S}} + p_{I(i),\mathcal{S}}p_{i+1,k}] \tag{4.29}$$

$$p_{I(i+1),\mathcal{S}} = \frac{1}{\kappa_{I(i+1)}}(p_{I(i),\mathcal{S}}p_{i+1,\mathcal{S}}) \tag{4.30}$$

for $k = 1, \dots, M$, $i = 1, \dots, R-1$, and $\kappa_{I(i+1)}$ is a normalizing factor defined by

$$\kappa_{I(i+1)} = \left[1 - \sum_{j=1}^{M} \sum_{\substack{k=1 \\ k \neq j}}^{M} p_{I(i),j}p_{i+1,k}\right] \tag{4.31}$$

Finally, we obtain m as $m_{I(R)}$. For the purpose of decision making, we now define a probability function P_m on $\mathcal{S}$ derived from m via the *pignistic transformation* as follows

$$P_m(c_k) = m(\{c_k\}) + \frac{1}{M}m(\mathcal{S}) \text{ for } k = 1, \dots, M \tag{4.32}$$

and we have the following decision rule:

$$j = \arg\max_{k} P_m(c_k) \tag{4.33}$$

It would be interesting to note that an issue may arise with the orthogonal sum operation, and is in using the total probability mass κ associated with conflict as defined in the normalization factor. Consequently, applying it in an aggregation process may yield counterintuitive results in the face of significant conflict in certain situations as pointed out in [Zadeh 1984]. Fortunately, in the context of the weighted combination of classifiers, by discounting all $P(\cdot|\mathbf{f}_i$ $(i = 1, \ldots, R)$ at corresponding rates $(1 - \alpha_i)$ $(i = 1, \ldots, R)$, we actually reduce conflict between the individual classifiers before combining them.

4.3.4 The Discounting-and-Averaging Combination Strategy

In this strategy, instead of using Dempster's rule of combination after discounting $P(\cdot|\mathbf{f}_i)$ at the discount rate of $(1 - \alpha_i)$, we apply the averaging operation over BPAs m_i $(i = 1, \ldots, R)$ to obtain the BPA m defined by

$$m(A) = \frac{1}{R}\sum_{i=1}^{R} m_i(A) \tag{4.34}$$

for any $A \in 2^{\mathcal{S}}$. By definition, we get

$$\begin{aligned} m(\{c_k\}) &= \frac{1}{R}\sum_{i=1}^{R} \alpha_i P(c_k|\mathbf{f}_i), \text{ for } k = 1, \ldots, M & (4.35)\\ m(\mathcal{S}) &= 1 - \frac{\sum_{i=1}^{R}\alpha_i}{R} \triangleq 1 - \overline{\alpha} & (4.36)\\ m(A) &= 0, \forall A \in 2^{\mathcal{S}} \setminus \{\mathcal{S}, \{c_1\}, \ldots, \{c_M\}\} & (4.37) \end{aligned}$$

Note that the probability mass unassigned to individual classes but the whole frame of discernment $\mathcal{S}$, $m(\mathcal{S})$, is the average of discount rates. Therefore, if instead of allocating the average discount rate $(1 - \overline{\alpha})$ to $m(\mathcal{S})$ as above, we use it as a normalization factor and easily obtain

$$\begin{aligned} m(\{c_k\}) &= \frac{1}{\sum_i \alpha_i}\sum_{i=1}^{R} \alpha_i P(c_k|\mathbf{f}_i), \text{ for } k = 1, \ldots, M & (4.38)\\ m(A) &= 0, \forall A \in 2^{\mathcal{S}} \setminus \{\{c_1\}, \ldots, \{c_M\}\} & (4.39) \end{aligned}$$

which interestingly turns out to be the weighted mixture of individual classifiers as defined in (4.21). Then we have the decision rule (4.22).

It should be worth noting that since the average discount rate $(1-\overline{\alpha})$ is a constant, the decision rule based on the weighted mixture of individual classifiers is the same as that based on the probability function P_m with m defined by (4.35)–(4.37) via the pignistic transformation.

4.4 Combination Based on OWA Operators

By considering individual classifiers as experts who have their own soft decisions (for instance, probability distributions over the set of classes) on the word sense identification, we now face with the problem how to derive a consensus decision based on their individual decisions. Intuitively, in such a situation, one (i.e., decision maker) may have a decision making strategy based on linguistic quantifiers, for instance "a consensus decision should be the one that *all* experts supported it". We have mimicked this decision making behavior of human beings for decision fusion in the context of WSD. In particular, we have used OWA operators for classifier fusion in their semantic relation to linguistic quantifiers [Zadeh 1983]. Under such a formulation, we will provide a framework for combining classifiers, which also yields several commonly used decision rules for WSD. In particular, some fuzzy majority voting have been derived, which correspond to the combination rules, such as Max rule, Min rule, and Median rule. It should be worth mentioning that the use of OWA operators in classifier combination has been studied, for example, in [Kuncheva 2001]. In this work, however, we use OWA operators for classifier fusion in their semantic relation to linguistic quantifiers so that we could provide a framework for combining classifiers, which also yields several commonly used decision rules but without some strong assumptions made in the work by Kittler et al. [Kittler *et al.* (1998)]..

4.4.1 OWA Operators

The notion of OWA operators was first introduced in [Yager 1988] regarding the problem of aggregating multi-criteria to form an overall decision function. A mapping

$$F : [0,1]^n \to [0,1]$$

is called an OWA operator of dimension n if it is associated with a weighting vector $W = [w_1, \ldots, w_n]$, such that 1) $w_i \in [0,1]$ and 2) $\sum_i w_i = 1$, and

$$F(a_1, \ldots, a_n) = \sum_{i=1}^{n} w_i b_i$$

where b_i is the i-th largest element in the collection $a_1, \ldots, a_n$.

As suggested by Yager [Yager 1988], there exist at least two methods for obtaining weights w_i's. The first approach is to use some kind of learning mechanism. That is, we use some sample data, arguments and associated aggregated values and try to fit the weights to this collection of sample data. The second approach is to give some semantics or meaning to the weights. Then, based on these semantics we can directly provide the values for the weights. In the following we use the semantics based on fuzzy linguistic quantifiers for the weights.

The fuzzy linguistic quantifiers were introduced by Zadeh in [Zadeh 1983]. According to Zadeh, there are basically two types of quantifiers: absolute, and relative. Here we focus on the relative quantifiers typified by terms such as *most*, *at least half*, *as many as possible*. A relative quantifier Q is defined as a mapping $Q : [0,1] \to [0,1]$ verifying $Q(0) = 0$, there exists $r \in [0,1]$ such that $Q(r) = 1$, and Q is a non-decreasing function. For example, the membership function of relative quantifiers can be defined [Herrera & Verdegay (1996)] as

$$Q(r) = \begin{cases} 0 & \text{if } r < a \\ \frac{r-a}{b-a} & \text{if } a \leq r \leq b \\ 1 & \text{if } r > b \end{cases} \tag{4.40}$$

with parameters $a, b \in [0,1]$.

Then, Yager [Yager 1988] proposed to compute the weights w_i's based on the linguistic quantifier represented by Q as follows:

$$w_i = Q(\frac{i}{n}) - Q(\frac{(i-1)}{n}), \text{ for } i = 1, \ldots, n. \tag{4.41}$$

4.4.2 OWA Operator Based Combination Scheme

Let us return to the problem of identifying the sense of a given word $\mathbf{w}$ as described above. As mentioned above, suppose that each individual classifier is based on an information source $\mathbf{f}_i$, which can be also considered as providing the information inspired by a semantical or syntactical criterion for the purpose of word sense identification. Let us assume that we have R classifiers corresponding to R information sources $\mathbf{f}_i$ of the context, each of which provides a soft decision for identifying the right sense of the target word $\mathbf{w}$ in the form of a posterior probability $P(c_k|\mathbf{f}_i)$, for $i = 1, \ldots, R$.

Under such a consideration, we now can define an overall decision function D, with the help of an OWA operator F of dimension R, which combines individual opinions to derive a consensus decision as follows:

$$D(c_k) = F(P(c_k|\mathbf{f}_1), \ldots, P(c_k|\mathbf{f}_R)) = \sum_{i=1}^{R} w_i p_i \tag{4.42}$$

where p_i is the i-th largest element in the collection $P(c_k|\mathbf{f}_1), \ldots, P(c_k|\mathbf{f}_R)$, and $W = [w_1, \ldots, w_R]$ is a weighting vector semantically associated with a fuzzy linguistic quantifier. Then, the fuzzy majority based voting strategy suggests that the target word $\mathbf{w}$ should be assigned to class c_j provided that $D(c_j)$ is maximum, namely

$$j = \arg\max_k D(c_k) \tag{4.43}$$

As studied in [Yager 1988], using Zadeh's concept of linguistic quantifiers and Yager's idea of associating their semantics to various weighting vectors W, we can obtain many commonly used decision rules as following.

Max Rule.

First let us use the quantifier *there exists* which can be relatively represented as a fuzzy set Q of $[0, 1]$ such that $Q(r) = 0$, for $r < 1/R$ and $Q(r) = 1$, for $r \geq 1/R$. We then obtain from (4.41) the weighting vector $W = [1, 0, \ldots, 0]$, which yields from (4.42) and (4.43) the Max Decision Rule as

$$j = \arg\max_k \left[\max_i P(c_k|\mathbf{f}_i)\right] \tag{4.44}$$

According to the quantifier-based semantics associated with this weighting vector [Yager 1988], the max decision rule states that a decision of labeling for example $\mathbf{e}$ should be made if *there exists* an individual classifier supports that decision. In other word, the label selection of an example is followed the decision of the best classifier or followed the decision that obtains the maximum degree of supports from all the individual classifier.

Min Rule.

Similarly, if we use the quantifier *for all* which can be defined as a fuzzy set Q of $[0, 1]$ such that $Q(1) = 1$ and $Q(r) = 0$, for $r \neq 1$ [Yager 1988]. We then obtain from (4.41) the weighting vector $W = [0, \ldots, 0, 1]$, which yields from (4.42) and (4.43) the Min Decision Rule as

$$j = \arg\max_k \left[\min_i P(c_k|\mathbf{f}_i)\right] \tag{4.45}$$

Also, according to the quantifier-based semantics associated with W, the min decision rule states that a decision of labeling for example $\mathbf{e}$ should be made if *all* individual classifiers support that decision.

Suppose that the i^{th} classifier supports label l_j a degree $d_{i,j}(\mathbf{e})$, then it means this classifier opposes a degree $(1 - d_{i,j}(\mathbf{e}))$ to l_j. For each support degree $d_{i,j}(\mathbf{e})$, let we name its corresponding opposition is $\overline{d'_{i,j}}(\mathbf{e})$, then the min rule can be rewritten as follow.

$$\hat{j} = \arg\min_{j=1\cdots c}\left[\min_{i=1}^{R}[1 - d_{i,j}(\mathbf{e})]\right] = \arg\min_{j=1\cdots c}\left[\min_{i=1}^{R}\overline{d'_{i,j}}(\mathbf{e})\right] \tag{4.46}$$

With this presentation of the min rule, the label selection of an example is followed the decision which minimums the opposition from all the individual classifiers.

Median Rule.

In order to have the Median decision rule, we use the absolute quantifier *at least one* which can be equivalently represented as a relative quantifier with the parameter pair $(0, 1)$ for the membership function Q in (4.40). Then we obtain from (4.41) the weighting vector $W = [1/R, \ldots, 1/R]$, which from (4.42) and (4.43) leads to the median decision rule as:

$$j = \arg\max_{k}\left[\frac{1}{R}\sum_{i=1}^{R} P(c_k|\mathbf{f}_i)\right] \tag{4.47}$$

Fuzzy Majority Voting Rules.

We now use the relative quantifier *at least half* with the parameter pair $(0, 0.5)$ for the membership function Q in (4.40). Then, depending on a particular value of R, we can obtain from (4.41) the corresponding weighting vector $W = [w_1, \ldots, w_R]$ for the decision rule, denoted by FM1, as:

$$j = \arg\max_{k}\left[\sum_{i=1}^{R} w_i p_i\right] \tag{4.48}$$

where p_i is the i-th largest element in the collection $P(c_k|\mathbf{f}_1), \ldots, P(c_k|\mathbf{f}_R)$.

Similarly, we can also use the relative quantifier *as many as possible* with the parameter pair $(0.5, 1)$ for the membership function Q in (4.40) to obtain the corresponding decision rule, denoted by FM2.

Interestingly also, from the following relation

$$\prod_{i=1}^{R} P(c_k|\mathbf{f}_i) \le \min_{i=1}^{R} P(c_k|\mathbf{f}_i) \le \sum_{i=1}^{R} w_i p_i \le \max_{i=1}^{R} P(c_k|\mathbf{f}_i) \le \sum_{i=1}^{R} P(c_k|\mathbf{f}_i) \tag{4.49}$$

it suggests that the Max and Min decision rules can be approximated by the upper or lower bounds appropriately on the Median rule, as presented in [Le *et al.* (2006a)].

4.5 Second-Layer Combination

In this section we present new schemes for classifier combination, in which one more phase of combination is added to the end of common combination strategies. This is motivated by an observation that different results yielded by applying various combination rules on the set of individual classifiers can be again used for one more combination. This new phase of combination is called second-layer combination, and thus the previous combination can be considered as the first-layer combination. This proposal is also motivated by that combination usually gives better result in comparison with individual classifiers, so the second-layer combination is hoped to yield better results in comparison with the first-layer combination.

Concerning classifier combination we face the problem that fixing a rule for every tests is not the best choice. In this case, models of second-layer combination can be a solution. For example, we can use the meta-vote on the outputs of these combination rules to which we can avoid the bias to any combination rule, and we can even obtain better results.

In the below, we introduce two models of second-layer combination, namely meta-combination and meta-stacking.

4.5.1 Meta-Combination

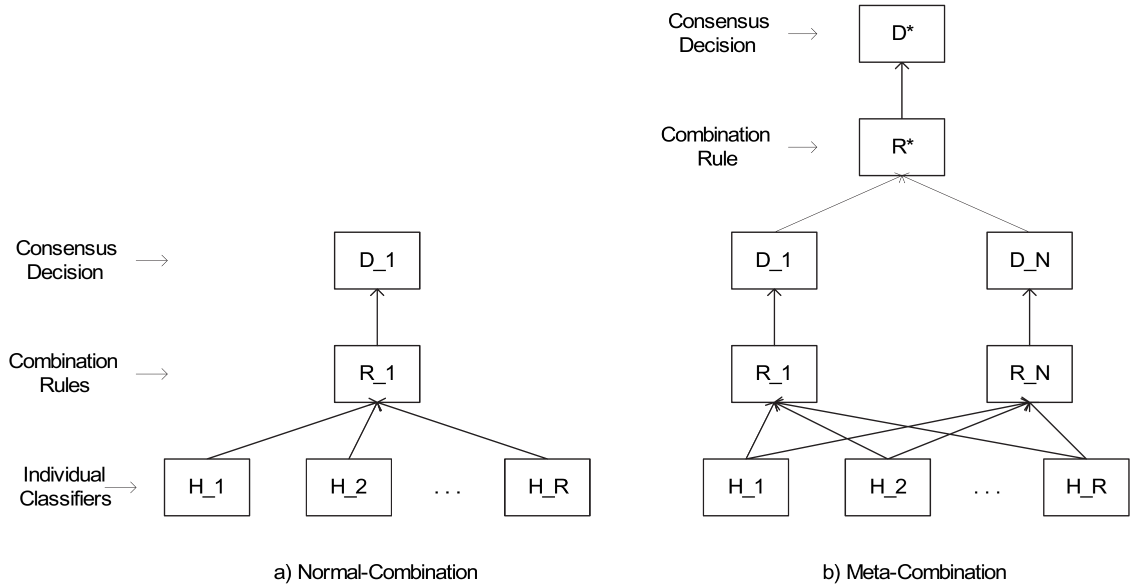


Figure 4.3: Combination Schemes

Fig.4.3 shows two schemes of combination, including the normal-combination scheme (also considered as a first-layer combination strategy) in part (a) and the meta-combination scheme (also considered as a second-layer combination strategy) in part (b). In the normal-combination , a combination rule is applied on the outputs of individual classifiers to yield the final decision. In the meta-combination scheme, we first use N different combination rules to generate N outputs, and then continue apply a combination rule on these N outputs to yield the final decision. This meta-combination scheme is hoped to improve accuracy of the normal-combination scheme. Beside that, it also can be a solution for the case when we are not sure which is the best among N combination rules used.

Note that we can always apply the majority voting rule in the second-layer combination. Beside that, if all combination rules at the first-layer can output probability distributions over classes, we can also apply other rules at the second-layer, such as median rule, max rule, min rule, etc. Reviewing related studies, we see that [Florian & Yarowsky (2002)] used median/average rule and majority voting at the second-layer, as shown in Table. 4.1.

4.5.2 Meta-Stacking Combination Models

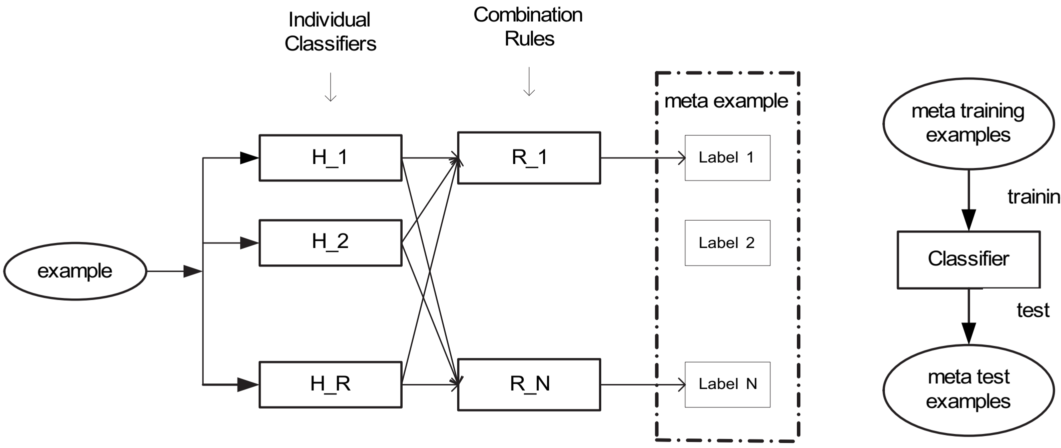


Figure 4.4: Meta-Stacking Combination Model

In the normal-stacking model, we rebuild the test and training examples such that each example is replaced by a sequence of labels which are the outputs of individual classifiers. After that, using these new training examples to train a classifier, and the test on the new test examples. The process of stacking was described in Fig. 4.2.

Start from the stacking model, instead of using outputs of the individual classifiers for generating new examples for training and test datasets, we use the outputs from applying combination rules on the outputs of individual classifiers, we then have a meta-stacking model of combination, as shown in Fig. 4.4. With an observation that a first-layer combination usually gives better results in comparison with results from individual classifiers, the meta-stacking combination using these results from the first-layer combination of stacking-model is hoped to make better performance of classification in comparison with stacking combination which directly uses the results from individual classifiers.

Reviewing related studies in word sense disambiguation, there are a few of them which use stacking methods such as [Klein *et al.* (2002), Wang & Matsumoto (2004)], and there are no study mentioned meta-stacking. In the experiment as below, these two kinds of stacking method will be compared with each other, and also compared with the directly use of combination rules.

4.6 Experiment

In this section, experimental results for all combination strategies mentioned above are shown, in which various combination rules and two combination schemes, the first-layer and the second-layer, are implemented on Senseval-2 and Senseval-3. In addition, we also these tests also implemented with two types of individual classifiers: one is based on different representations of context (or different feature sets), and the other is based on different machine learning algorithms.

4.6.1 Generation of Individual Classifiers

Based on Different Representations of Context

The idea of using different feature sets to build individual classifiers comes from the observation that various ways of using the context could be considered as providing different information sources to identify the meaning of the target word. The various kinds of features usually used for identifying word senses include bags of content words, collocations, or some relationship between the target word with surrounding words such as syntactic relation and distance relation, each of them can be represented by a feature subset. Combining all these feature subsets in a unique set is not always the best choice because each of them, even those of the same kind (for example bag of content words) but with different window sizes, has a different impact on the meaning of the polysemous word, depending on a particular context or on the target word itself. This intuitive observation prompted us to use multi-representation of context as a means of combining individual decisions to reach a consensus. Therefore, a appropriate way to build individual classifiers for a combination strategy is that these classifiers are built based on different representations of context.

In Chapter 4, we have investigated 8 feature sets (corresponding different kinds of information) and proposed a feature selection method to find useful features. Finally, 7 feature sets have been selected, including F_1^a, F_1^b, F_1^c, F_2, F_3, F_5, F_6 (see Chapter 4 for more detail). Among these sets, F_6 contains features about syntactic information. However, in some contexts of the target word, we can not extract syntactic information due to the incomplete information obtained from the parser. In the other hand, ordered part-of-speech tags, as contained in feature set F_5, can be considered as a kind of syntactic information. This suggests us to combine these two feature set into a unique set which can represent for the syntactic information. Particularly, we design 6 different sets of features (can be seen as 6 views of context), they include $V_1 = F_1^a$; $V_2 = F_1^b$; $V_3 = F_1^c$; $V_4 = F_2$; $V_5 = F_3$; $V_6 = F_5 \cup F_6$. Using a supervised learning algorithm on these six feature sets, we will have six individual classifiers.

Based on Different Algorithms

In order to build individual classifiers based on different algorithms, we use three supervised learning algorithms including NB, MEM, and SVM. The feature set used is the same for all the three algorithms, they include the whole features from all the seven feature sets F_1^a, F_1^b, F_1^c, F_2, F_3, F_5, F_6 (i.e. for the six views, V_1, V_2, V_3, V_4, V_5, V_6). Call the overall feature set is **F**, it is defined as:

$$\mathbf{F} = F_1^a \cup F_1^b \cup F_1^c \cup F_2 \cup F_3 \cup F_5 \cup F_6$$
$$= \bigcup_{i=1}^{6} V_i$$

Table 4.2: Combination with different feature sets

	Best Ind.	DS1	DS2	NB	Max	Min	Median	FM1	FM2	Vote	WVote	Meta Vote
Sen2 +NB	56.8	65.0	64.8	64.0	62.2	62.4	**65.4**	65.1	64.7	63.0	64.8	***65.7***
Sen2 +MEM	59.0	61.4	60.2	61.4	58.2	58.6	**63.4**	63.3	62.6	62.0	63.0	*63.0*
Sen2 +SVM	57.8	59.9	58.1	**61.6**	59.8	59.8	59.7	59.3	59.6	58.1	58.7	*60.1*
Sen3 +NB	64.1	**72.4**	**72.5**	72.0	69.3	70.3	72.2	71.7	72.2	69.7	71.6	***72.5***
Sen3 +MEM	64.5	66.6	63.3	66.7	64.8	64.2	68.9	68.8	69.0	68.1	68.8	***69.1***
Sen3 +SVM	65.3	67.3	76.1	67.6	**67.9**	67.8	67.1	67.4	66.8	64.8	66.3	*67.4*

4.6.2 Experimental Results and Discussion

Combination Rules and Meta-Voting

Table. 4.2 shows the results of applying combination rules on individual classifiers, in which these individual classifiers are built by using a supervised learning algorithm trained on different feature sets. This experiment are implemented for lexical sample tasks of Senseval-2 and Senseval-3. In addition, we also tried three supervised learning algorithms, including NB, MEM, and SVM. In this table, the column named "Best Ind." denotes the best result from outputs of individual classifiers; for other columns standing for other combination rules, DS1 and DS2 respectively stand for Dempster-Shafer with and without discount factor, NB stands for Naive Bayes, Vote stands for majority voting, and MVote stands for weighted voting. Note that DS1 is also the product rule. In this experiment we also use the majority voting on the outputs of all combination rules in the first-layer combination, and this combination strategy is called meta voting. The reason for choosing only majority voting for second-layer combination is that this rule just requires each combination rule of the first-layer combination output the best label for each test. The result of meta-voting is placed at the last column.

Table 4.3 also shows the results of a similar test as of the Table 4.2 but instead of using different feature sets, we built three individual classifiers based on three different learning algorithms, including NB, MEM, and SVM. These algorithms and they are trained on the same feature set, **F**, as mentioned above.

Through results from these two tables, we can extract the following remarks.

- Applying combination rules can improve performances of individual classifiers. Specially in the case of using different feature sets, the combinations can increase accuracy of the best individual classifier up to about 8% (from 56.8% to 65.4%, and 64.1% to72.4%).

Table 4.3: Combination with different learning methods

	Best Ind.	DS1	DS2	NB	Max	Min	Median	FM1	FM2	vote	wvote	meta vote
Sen2	64.8	66.1	65.4	65.7	65.5	65.8	66.3	66.3	**66.4**	65.8	66.0	***66.4***
Sen3	72.0	73.3	73.2	73.2	73.1	73.1	73.6	73.6	73.7	73.5	**73.8**	***73.6***

Table 4.4: Meta-Voting on the two sets of individual classifiers: using different feature sets and different learning methods

	meta voting
Senseval2	66.3
Senseval3	73.8

- The best combination rule is changed depending on each particular dataset.
- Comparing the combinations in two kinds of generation of individual classifiers, we see that using the set of the whole features with different learning algorithms gives better results.
- In most cases, results of meta-voting can be comparable to the best results, see Fig. 4.2 and Fig. 4.3.

Through these two tables, we can see that using classifier combination much improves accuracies of individual classifiers, both on the two strategies of generating individual classifiers. It also shows that the best combination rule is not fixed through different datasets, thus we face to the question: which combination rule should be chosen? In this case, the meta-voting can be an appropriate choice.

In order to investigate the effectiveness of using meta-combination on the whole outputs from both two types of individual classifiers (based on different feature sets and based on different machine learning algorithms) we also did a corresponding experiment, and the result is shown in Table 4.3. This results does not improve accuracy while takes more cost in computation.

Stacking and Meta-Stacking

In the following, we carry out some tests on the stacking approach, in which two models of combination including first-layer stacking and second-layer stacking were implemented.

Table 4.5: Stacking with individual classifiers based on different feature sets

	Best	First-Layer Stacking			Second-Layer Stacking		
		NB	MEM	SVM	NB	MEM	SVM
Senseval2	56.8	**62.8**	62.4	60.2	**65.7**	65.5	64.6
Senseval3	64.1	**69.4**	68.0	67.5	**72.5**	71.4	71.9

Table 4.6: Stacking with individual classifiers based on different learning methods

	Best	First-Layer Stacking			Second-Layer Stacking		
		NB	MEM	SVM	NB	MEM	SVM
Senseval2	64.8	**65.5**	65.5	65.0	**66.2**	**66.2**	65.9
Senseval3	72.0	**73.4**	72.6	72.5	73.5	**73.6**	**73.6**

Table 4.7: Combination of Different Feature Sets and Different Learning Methods in Stacking methods

	First-Layer Stacking			Second-Layer Stacking		
	NB	MEM	SVM	NB	MEM	SVM
Senseval2	**65.9**	65.4	64.0	**66.3**	**66.3**	65.4
Senseval3	**72.8**	72.1	72.2	**73.7**	73.2	73.2

We also used both sets of individual classifiers, one is created by using different feature sets and NB algorithm (we chose NB because of its effectiveness in the before experiments), and the other is created by using different learning algorithms (NB, MEM, and SVM) on the whole features. Note that, after the train and test examples are rebuilt by using the outputs of individual classifiers, we did three tests corresponding to using three supervised learning algorithms including NB, MEM, and SVM. The results of these tests are shown in Table 4.5 and Table 4.6, in which the column denoted by "Best" shows results of the best individual classifier. And from these tables, we can extract some remarks as follows.

- Stacking methods can improve the performance of individual classifiers.
- Results of the second-layer stacking models are better than results of the first-layer stacking models. In most cases, the best results are obtained when using NB as the supervised learning algorithm at the final step of stacking methods, in comparison with using MEM and SVM algorithms.
- In stacking models, concerning the generation of individual classifiers, using different learning algorithms gives better results in comparison with using different feature sets.

Similar to a test in meta-voting, in order to investigate the effectiveness of using more individual classifiers, we also carry out the tests for the first-layer stacking and the second-layer stacking models with the individual classifiers taken from both types: based on different machine learning algorithms and different feature sets. Results are shown in Table 4.7. Comparing with the corresponding results in Table 4.5 and Table 4.6, we can see that such way of using more individual classifiers gives slightly better results.

For an intuitively view, we summarize the results from applying various combination strategies in Fig. 4.5 (test on Senseval-2) and Fig.. 4.6 (test on Senseval-3). In these two figures, x-axis holds for the types of combination in which T_1 denotes normal combination rules, T_2 denotes stacking, T_3 denotes meta-voting, and T_4 denotes stacking in second-layer, y-axis holds for the accuracies. For each type of combination, we tried two sets

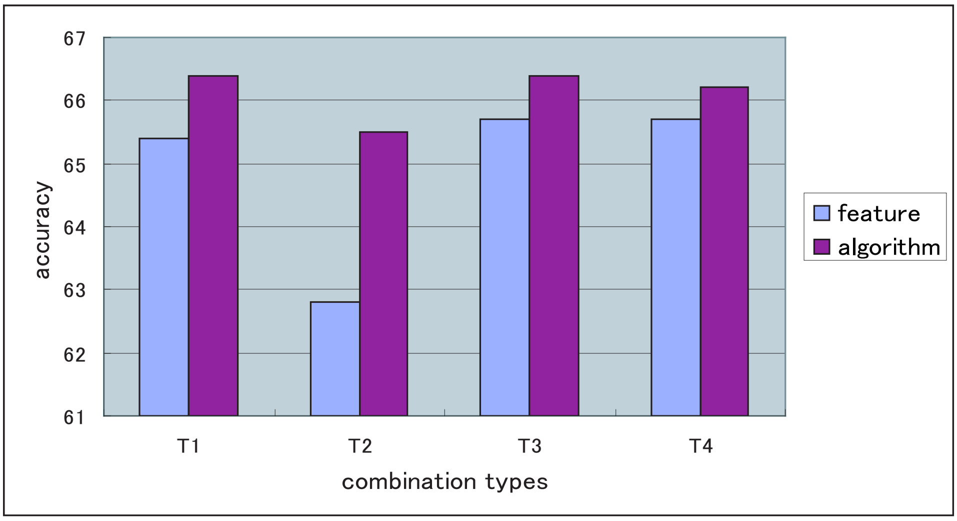


Figure 4.5: Test on Senseval-2: An overview of the best results of different types of combination

of individual classifiers, one is based on different feature sets, and the other is based on different machine learning algorithms. The column named “feature” stands for the first type of individual classifiers, and the column named “algorithm” stands for the second type of individual classifiers.

These two figures intuitively show again some conclusions as extracted above, which include:

- To generate individual classifiers for combination strategies, using different machine learning algorithms is more effective than using different feature sets.
- Though using first-layer combination gives the best results, it is still suggested to use meta-voting on the outputs of those combination rules, that is because the best rule is changed through different datasets.
- Stacking does not show the effectiveness in comparison with other strategies, however, second-layer stacking method can be also comparable with the best strategies.

Table 4.8 shows a comparison between our proposal, the meta-voting on different learning methods, and previous studies. It shows that the obtained results from our proposal are just lower than the best result in the current study, i.e. [Ando(2006)], while better than others.

4.7 Summary

This chapter presents one of the three main contents in our thesis, that is the problem of applying classifier combination for WSD. In this work, some strategies of classifier combination from previous studies have been investigated. Two new approaches of classifier combination for WSD, which are based on Dempster-Shafer theory of evidence and OWA

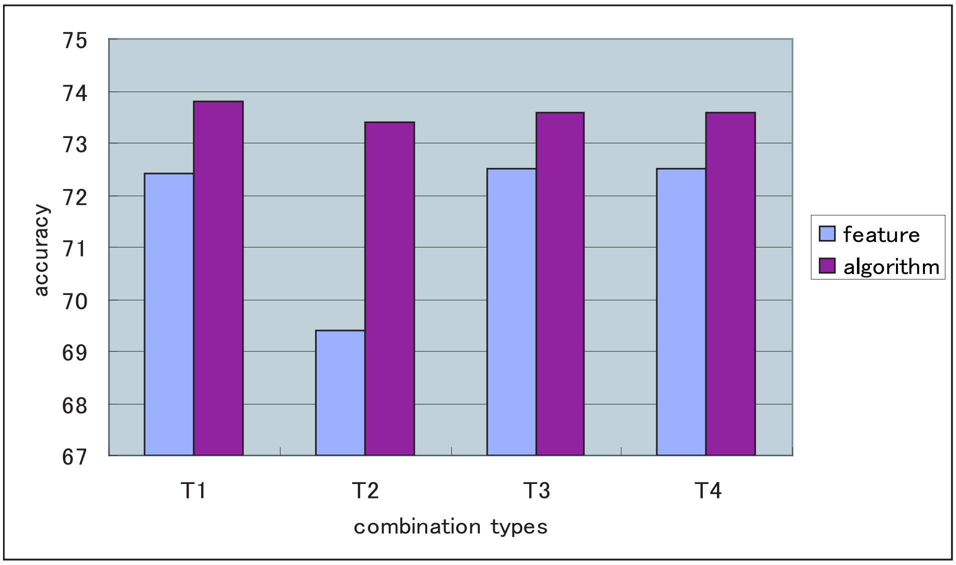


Figure 4.6: Test on Senseval-3: An overview of the best results of different types of combination

operators have been presented. The second-layer combination strategies were also proposed, and various experiments were done. Particularly, we have made the contributions as follows.

- We have stated a classification problem, particularly for WSD, by the concepts of Dempster-Shafer (D-S) theory of evidence [Shafer (1976)]. This can be considered as a new framework for WSD problem. Applying the Dempster rule of combination (it is also considered as the orthogonal sum) on this framework, we achieve a new combination rule called D-S combination rule. Furthermore, we also applied the discount operator on this framework to derive Discounting-and-Orthogonal sum rule. Under this combination strategy, we also can yield other combination rules, such as Average rule and Discounting-and-Average rule.

- We have applied Ordered Weighted Aggregating (OWA) operators [Yager 1988] and linguistic quantifiers to the problem of classifier combination. As the result, some fuzzy majority voting have been derived, which correspond to the combination rules, such as Max rule, Min rule, and Median rule. The use of fuzzy linguistic quantifiers not only help deriving but also provides a human-like interpretations to these rules. Note that, some of these combination rules are also yielded in the work of [Kittler *et al.* (1998)], but with some strong assumptions which is difficult to be accepted in WSD problem. Moreover, this approach does provide us a clear interpretation about the semantic of these combination rules.

- We have proposed two second-layer combination strategies, called meta combination and meta-stacking. This proposal can be considered as the generalization of some special cases in previous studies, for example [Florian & Yarowsky (2002), Florian *et al.* (2002)] used the average and voting rules on the outputs of the first-layer combination rules.

- Two kinds of generating individual classifiers were investigated, one is based on different feature sets and the other is based on different machine learning algorithms. Most previous studies just considered the second method.

Table 4.8: Comparison with previous studies on Senseval-2 and Senseval-3

Method	Senseval-2	Senseval-3
	accuracy(%)	accuracy(%)
our method: meta-voting	**66.4**	**73.6**
ASO multi-task, [Ando(2006)]	*68.1* optimized parameters	*73.8*
classifier combination, [Florian & Yarowsky (2002)]	*66.5*	
polynomial KPCA, [Wu *et al.* (2004)]	65.8	
SVM, [Lee & Ng(2002)]	65.4	
The best systems in Senseval contests	64.2	72.9
Baseline	48.35	55.17

- We have carried out various tests. The obtained results show that the meta-voting strategy based on various combination rules in which individual classifiers based on different machine learning algorithms gives the best results. These results can be comparable with the best systems when testing on English lexical sample tacks of Senseval-2 and Senseval-3.

Chapter 5

Semi-Supervised Learning

This chapter presents our work on exploiting unlabeled data to improve performance of supervised-learning based WSD. The process of using both labeled and unlabeled data to build a classifier is called *semi-supervised learning*, and the approach in which labeled data is iteratively extended will be investigated. We first explicitly identify some problems occurring in this approach (particularly on self-training and co-training algorithms) and then propose solutions for them. Consequently, a new semi-supervised learning algorithm with several variants are generated. To test the effects of the proposed solutions, we also develop various models of the new bootstrapping algorithm and tested them on Senseval-2 and Senseval-3. Further, we propose a method that can combine advantages from the use of classifier combination techniques (as presented in the chapter 4) and the use of unlabeled data.

5.1 Introduction

5.1.1 Methods in Semi-Supervised Learning

The semi-supervised learning is a special form of classification in which it uses both labeled and unlabeled data while a traditional classifier uses only labeled data. Labeled examples however are often difficult, expensive, or time consuming to obtain, as they require efforts of experienced human annotators. Meanwhile unlabeled examples may be relatively easy to collect, but there has been few ways to use them. Semi-supervised learning addresses this problem by using large amount of unlabeled data, together with the labeled data, to build better classifiers. Semi-supervised learning methods use unlabeled data to either modify or re-prioritize hypotheses obtained from labeled data alone. In our opinion, semi-supervised learning methods can be grouped into two approaches as follows.

Methods Based on Optimizing Parameters

In the first approach the learners try to optimize parameters of the classification model using both labeled and unlabeled data. [Miller & Uyar (1997)] and [Nigam *et al.* (2000)] used a generative model for the classifier and used Expectation Maximization to estimate the model's parameters trained on both labeled and unlabeled data. [Joachims (1999)] used transductive inference for support vector machines to optimize performance on a specific test set. While [Blum & Chawla (2001)] used a graph-based method in which a graph based on the whole examples is first constructed and then the minimum cut on the graph yields an optimal labeling for the unlabeled examples according to a certain optimization function. Two of the most popular methods in this approach are EM-based method [Nigam *et al.* (2000)] and transductive SVM [Vapnik (1998)].

Transductive SVM

Transductive SVM is an extension of standard support vector machines with unlabeled data. In a standard SVM, only the labeled data is used, and the goal is to find a maximum margin linear boundary in the Reproducing Kernel Hilbert Space. In a transductive SVM, the unlabeled data is also used, and the goal is to find a labelling for the unlabeled data, such that a linear boundary has the maximum margin on both the original labeled data and the (now labeled) unlabeled data. The decision boundary has the smallest generalization error bound on unlabeled data. Although some success has been reported (e.g., see [Joachims (1999)], there has also been criticism pointing out that this method may not behave well under some circumstances [Zhang & Oles (2000)].

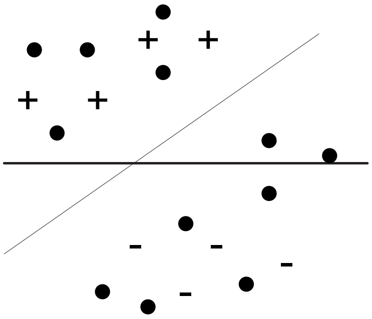


Figure 5.1: The maximum margin hyper-planes for Transductive Support Vector Machines

The relation between support vector machines and transductive support vector machines on the task of maximizing margin is shown in Fig. 5.1, in which positive/negative examples are marked as +/-, test examples as dots. The solid line is the solution of the inductive SVM, and the dashed line shows the transdutive classification.

EM-Based Algorithm

Some methods in semi-supervised learning are based on EM algorithm. The most well-known method was presented by [Nigam *et al.* (2000)], in which the model's parameters are iteratively updated by using the current model to infer (a probability distribution on) labels for the unlabeled data and then adjusting the model parameters to fit the (distribution on) filled-in labels. When the model defines a joint probability distribution over labeled data and unlabeled data, each iteration of the EM algorithm can be shown to increase the probability of the labeled data given the model parameters. However, EM is often subject to local minima situations in which the filled-in data and the model parameters fit each other well but the model parameters are far from their maximum-likelihood values. Furthermore, even if EM does find the globally optimal maximum likelihood parameters, a model with a large number of parameters will over-fit the data.

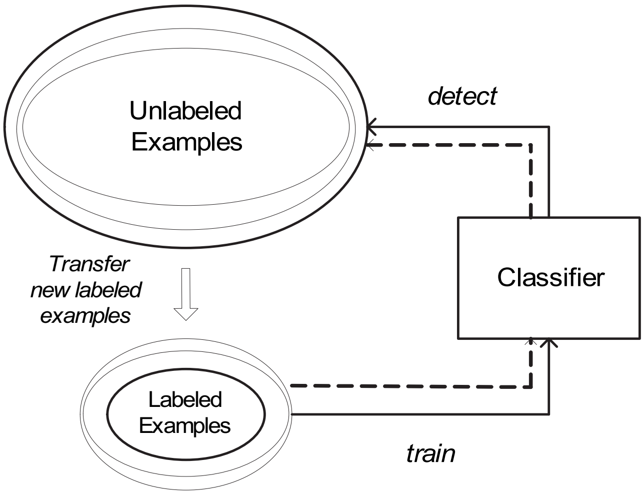


Figure 5.2: A Scheme to Describe the Process of Iteratively Extending Labeled Data

Methods Based on Iteratively Extending Labeled Data

In the second approach, learners follow a strategy in which the initial labeled data is iteratively extended, and finally a larger labeled data is obtained and used to generate the final classifier. From the literature review, we observe that a common method for enlarging labeled data is to use the classifier trained on the current labeled dataset to detect labels for unlabeled examples. Among those new labeled examples, some highly accurate ones are selected and added to the current labeled dataset. This process is iteratively repeated until there is no unlabeled example left, or until the number of iterations reaches a predefined threshold. Two well-known methods based on this approach are self-training [Yarowsky (1995)] and co-training [Blum & Mitchell (1998)]. Note that in self-training the classifier uses its own predictions to teach itself. This process of extending labeled data and retrain the classifier is also called self-teaching or bootstrapping. Self-training in [Yarowsky (1995)] is different from co-training in [Blum & Mitchell (1998)] is that self-training considers the overall feature space in one view while co-training splits the overall feature space into two views (see the next sections for more detail).

Our choice

In this work we follow the second approach because it seems intuitively suitable for WSD. As mentioned before, the right sense of a polysemous word in a certain context is determined based on the features extracted from this context. It means that some features can determine one sense, so we can use these features to detect new contexts for the polysemous word, which can again be used for extracting new features. Fig. 5.3 taken from [Yarowsky (1995)] illustrates an example of extending labeled data in which the part (a) shows the initial contexts and part (b) shows extended labeled contexts of the polysemous word "plant". As shown in this figure, at the beginning there are some initial training contexts which contain seed collocations (i.e. features) including *life* for label "A" and *manufacturing* for label "B"(as shown in part (a)), and then use these training contexts to detect new contexts (as shown in part (b)) which contain new seed features including *animal*, *cell*, *employee*, etc. The new contexts are then used as training contexts to detect

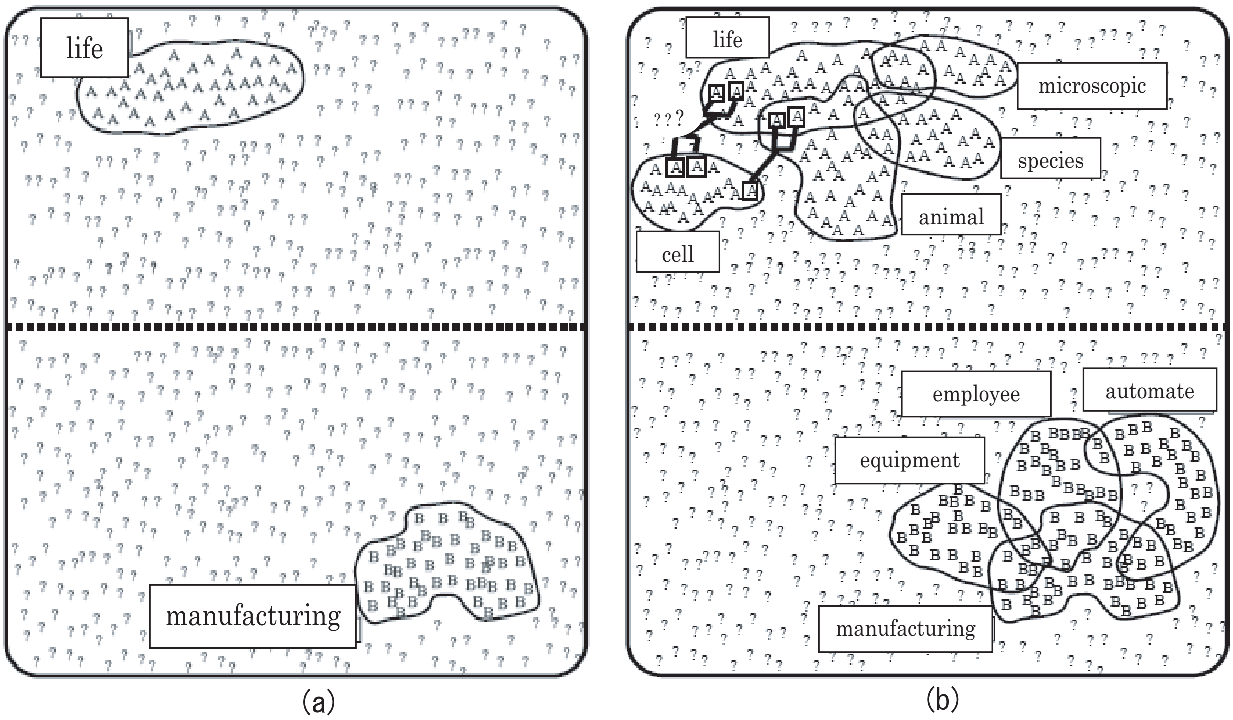


Figure 5.3: An Example of Extending Labeled Data in WSD

new features. This process can be repeated to recognize more new contexts, i.e. to enlarge training dataset. Note that some new features may appear just after several extension of training contexts, not at the first extension.

5.1.2 Problems and Motivations

As mentioned above, we follow the second approach in which the labeled data is iteratively extended. This process is described in the Algorithm 3 which can be considered as the general bootstrapping algorithm.

We now explicitly identify three problems (subtasks) in the general framework of semi-supervised learning which, according to our observation, may affect the performance of semi-supervised learning systems in practical applications, particularly in WSD. These problems are presented in detail as follows.

The first problem, denoted as P_1, regards the imbalance of labeled (training) data. We observe that if a classifier is built based on training data with a bias on certain classes (i.e., one or several classes dominate others), then this bias may become stronger at each extension of the labeled dataset. This is because a classifier tends to detect examples of dominant classes with high confidence, and consequently these examples are prioritized for a new set of labeled examples. Through steps of extending labeled data, the imbalance of labeled data is increased, which may result in decreasing the accuracy of the initial classifier. Previous studies just solved this by fixing the number of new labeled examples for each class, such as in [Blum & Mitchell (1998), Pierce & Cardie (2001)]. However, this can not be implemented in some certain circumstances, for example in the case when we can not achieve enough new labeled examples of a class for the corresponding number which is pre-defined. To tackle this problem we will provide a procedure which can flexibly retain class distribution and avoid fixing the number of new labeled examples. In addition, the effects of using or not using this solution for this problem will be shown through experiments on Senseval-2 and Senseval-3. Further, in order to ensure that retaining class distribution will not prevent to obtain an extended labeled data which has a more reliable class distribution, we assume that the original labeled data has the same class distribution as in "real" data.

The second problem, denoted by P_2, is that of how to determine a subset of new labeled examples with high confidence. It is clear that adding a large number of misclassified examples into the labeled dataset will probably result in generating a poor classifier in the end. Therefore, one aims at obtaining new labeled examples with the highest accuracy possible. To reach this target, previous studies normally used the so-called threshold-based selection of new labeled examples. In particular, given a new example which is assigned a label with a probability of detection, a threshold value for this probability is predefined to decide whether a new labeled example will be selected or not, such as in [Yarowsky (1995), Blum & Mitchell (1998), Collins & Singer (1999)]. However, this threshold-based method of selection may lead to a situation where choosing a higher threshold will create difficulty in extending labeled data, while it does not always result in correct classification. By contrast, a lower threshold may result in more misclassified examples, but allows more new labeled examples to be added. Therefore, the determination of a "correct" threshold in the approach becomes an important issue. In addition, determining a commonly used threshold for all unknown data is also inappropriate. To address these issues, we propose a method that flexibly and dynamically chooses an appropriate value for the threshold based on estimating the upper bound of the classification error rate of the obtained labeled dataset. Moreover, based on the observation that combining classifiers usually decrease the classification error rate, we aim at using different supervised learning algorithms to generate different classifiers and then combine them under a combination rule to increase the confidence of new labeled examples.

The third problem, denoted by P_3, is that of how to generate the final classifier when the process of extending labeled data is completed. According to the framework depicted in Algorithm 3, this process will be stopped when the number of iterations reaches a pre-specified value, or when the unlabeled dataset becomes empty. Normally, the classifier built on the labeled data obtained at the last iteration is chosen as the final one. Some studies use a development dataset to find the most appropriate value for the number of iterations, such as in [Pham *et al.* (2005), Mihalcea (2004)]. As mentioned in problem P_2, the last classifier may be built based on new training data with some misclassified examples, so both advantages and disadvantages are concurrently brought to the last classifier. Thus, choosing the classifier trained on the last labeled dataset as the final classifier is not always be a good solution. This observation suggests that we should combine the classifiers, which are obtained at each extension of labeled data, under classifier combination strategies to utilize advantages of these different classifiers.

For an intuitively view, we show the three problems as in Fig. 5.4.

By reviewing various related studies, especially regarding the WSD problem, we found that previous studies have not paid adequate attention to these three problems. In this thesis, we consider simultaneously these three problems for the objective of improving semi-supervised learning, particularly on self-training and co-training algorithms. Experiments are carried out on the English lexical sample tasks of Senseval-2 and Senseval-3.

Algorithm 3 – The General Bootstrapping Algorithm

Input: L (labeled data); U (unlabeled data); $k = 0$; K is the maximum number of iterations

Output: H – the final classifier

```
repeat
    $k \leftarrow k + 1$
    generate classifier $h$ trained on $L$
    use $h$ to label $U$, and obtain a labeled dataset $U_L$
    get $L' \subset U_L$ consisting of highly accuracy examples
    $L \leftarrow L \cup L'$; $U \leftarrow U \setminus L'$
until $U = \emptyset$ or $k > K$
use $L$ to generate the final classifier $H$
```

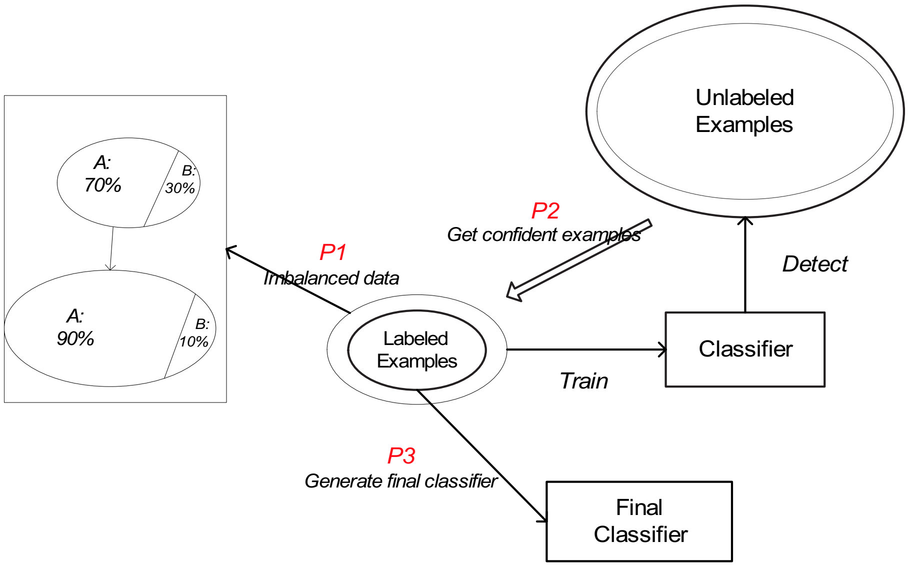


Figure 5.4: A View of Three Problems of the Process of Iteratively Extending Labeled Data

The obtained results show the effectiveness of the proposed solutions with a significant improvement of accuracy in comparison with supervised learning.

5.1.3 Semi-Supervised Learning for WSD: Related Work

Semi-supervised methods for WSD are characterized in terms of exploiting unlabeled data in learning procedures with the requirement of predefined sense inventory for target words. With only a small number of labeled examples, we may face two problems: firstly we cannot achieve the correct probability distribution over the feature space; secondly we may face the problem of lack of features in the training data, i.e. we may meet new features in test data. There were several studies which aimed to overcome the second problem by using external resources as, e.g., thesaurus or lexicons to disambiguate word senses or automatically generate sense-tagged corpus such as in [Lesk (1986), Lin (1997), McCarthy *et al.* (2004), Seo *et al.* (2004), Yarowsky (1992)] or by using similarity between words based on untagged corpus such as in [Karov & Edelman (1998)], or by basing on the WordNet resource such as in [Leacock (1998)]. Another approach when lacking labeled sense-tagged examples is to use a second language. In this approach, the studies, such as [Brown *et al.* (1991), Dagan & Itai (1994), Diab & Resnik (2002), Li & Li, Ng *et al.* (2003)], exploit the differences between mapping of words to senses in different languages with the help of bilingual corpora (e.g. parallel corpora or untagged monolingual corpora in two languages).

Other studies use bootstrapping (semi-supervised learning) algorithms to extend labeled data. Among them, Yarowsky algorithm [Yarowsky (1995)] is considered as the first bootstrapping algorithm. In this algorithm, the author use some labeled examples as seeds and extract from them the decision rules, which are then used to detect senses for new examples. This method is implemented with the help of the principle "one sense per collocation". As the result, we obtain new labeled examples added them to the current labeled dataset. New decision rules are continuously extracted and this process is repeated until converged. [Mihalcea (2004)] investigate the uses of co-training and self-training algorithms in word sense disambiguation, in which three parameters of these algorithms are discussed, including: number of iterations, number of examples selected from the unlabeled set for annotation at each iteration, and number of the labeled examples that are added at each iteration. In this paper, various settings of these parameters were tried on the development dataset to find the best setting. In addition, [Mihalcea (2004)] proposes a smoothing technique which uses majority voting on obtained individual classifiers instead of using only the current classifier at each iteration for extending labeled data. [Zheng (2005)] applies the Label Propagation based semi-supervised learning algorithm proposed by [Zhu & Ghahramani (2002)] to WSD. [Pham *et al.* (2005)] applies an algorithm which used co-training in spectral graph transductive(SGT) [Joachims (2003)] for WSD.

Reviewing related studies followed semi-supervised learning approach, we see that these three problems, P_1, P_2, and P_3 have not received adequate considerations. In this chapter, we will propose a new bootstrapping algorithm based on providing solutions for these problems. We also implement various experiments to show the effectiveness of the proposed solutions.

5.2 Self-Training and Co-Training

5.2.1 Co-Training

The co-training paradigm applies when accurate classification hypotheses for a task can be learned from either of two sets of features of the data, each called a view. The intuition behind Blum and Mitchell's co-training algorithm [Blum & Mitchell (1998)] is that two views of the data can be used to train two classifiers that can help each other. Each classifier is trained using one view of the labeled data, and then used to predicts labels for unlabeled examples. The most confident predictions of each classifier are selected and adding the corresponding examples with their predicted labels to the other's available training data. According to Blum and Mitchell [Blum & Mitchell (1998)], the condition for co-training algorithm works exactly is that the two views have to satisfy the two assumptions: firstly each view itself is sufficient to label task (or is sufficient to build a good classifier); the second is independent assumption of the two views. In the original definition of co-training, Blum and Mitchell [Blum & Mitchell (1998)] state conditional independence of the views as a required criterion for co-training to work.

In particularly, suppose that each example is represented by a feature vector x drawn from a set of possible values (an instance space) X. The task is to learn a classification function $f : X \rightarrow Y$ where Y is a set of possible labels. The characteristics of co-training can be described as follows.

- The features can be separated into two types: $X = X_1 \times X_2$ where X_1 and X_2 correspond to two different views of an example. In the named entity task, X_1 might be the instance space for the spelling features, X_2 might be the instance space for the contextual features. By this assumption, each element $x \in X$ can also be represented as $(x_l, x_2) \in X_1 \times X_2$.
- Each view of the example is sufficient for classification. That is, there exist functions f_l and f_2 such that for any example $x = (x_l, x_2)$, $f(x) = f_l(X_l) = f_2(x_2)$. We never see an example $x = (x_l, x_2)$ in training or test data such that $f_l(x_l) \neq f_2(x_2)$.

Thus the method makes the fairly strong assumption that the features can be partitioned into two types such that each type alone is sufficient for classification (x_l and x_2 are not correlated too tightly). Now assume we have n pairs $(x_{l,i}, x_{2,i})$ drawn from $X_1 x X_2$, where the first m pairs have labels y_i, whereas for $i = m+1, \ldots, n$ the pairs are unlabeled. In a fully supervised setting, the task is to learn a function f such that for all $i = 1, \ldots, m$, $f(x_{l,i}, x_{2,i}) = y_i$. In the co-training case, [Blum & Mitchell (1998)] argues that the task should be to induce functions f_l and f_2 such that:

1. $f_1(x_{1,i}) = f_2(x_{2,i}) = y_i$, for $i = 1, \ldots, m$
2. $f_1(x_{1,i}) = f_2(x_{2,i})$, for $i = m+1, \ldots, n$

So f_l and f_2 must (1) correctly classify the labeled examples, and (2) must agree with each other on the unlabeled examples. The key point is that the second constraint can be remarkably powerful in reducing the complexity of the learning problem. [Blum & Mitchell (1998)] gives an example that illustrates just how powerful the second constraint can be. Consider the case where $|X_1| = |X_2| = N$ and N is a "medium" sized

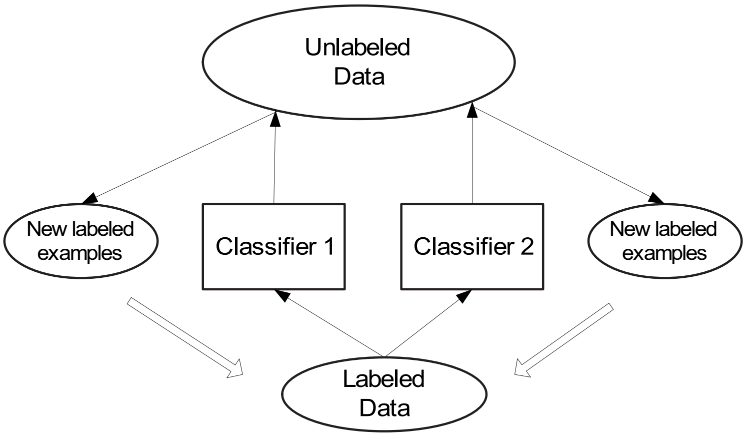


Figure 5.5: A Scheme for Co-Training Algorithm

number so that it is feasible to collect $O(N)$ unlabeled examples. Assume that the two classifiers are "rote learners": that is, f_l and f_2 are defined through look-up tables that list a label for each member of X_1 or X_2. The problem is a binary classification problem. The problem can be represented as a graph with $2N$ vertices corresponding to the members of X_1 and X_2. Each unlabeled pair $(x_{l,i}, x_{2,i})$ is represented as an edge between nodes corresponding to $x_{l,i}$ and $x_{2,i}$ in the graph. An edge indicates that the two features must have the same label. Given a sufficient number of randomly drawn unlabeled examples (i.e., edges), we will induce two completely connected components that together span the entire graph. Each vertex within a connected component must have the same label – in the binary classification case, we need a single labeled example to identify which component should get which label.

The original co-training algorithm [Blum & Mitchell (1998)] is presented in Algorithm 4, and a corresponding scheme is presented in Fig. 5.5.

Algorithm 4 – The Original Co-Training Algorithm

Input: a set L of labeled training examples; a set U of unlabeled examples; K is the number of iteration;

```
Create a pool $U'$ of examples by choosing $u$ examples at random from $U$
$k \leftarrow 0$
repeat
    $k \leftarrow k + 1$
    use $L$ to train a classifier $h_1$ that considers only the $x_1$ portion of $x$
    use $L$ to train a classifier $h_2$ that considers only the $x_2$ portion of $x$
    allow $h_1$ to label $p$ positive and $n$ negative examples from $U'$
    allow $h_2$ to label $p$ positive and $n$ negative examples from $U'$
    add these self-labeled examples to $L$
    randomly choose $2p + 2n$ examples from $U$ to replenish $U'$
until $k > K$
```

5.2.2 Self-Training

Self-training algorithm, also called Yarowsky algorithm [Abney (2004)] is one of the first bootstrapping algorithms to become widely known in computational linguistics. This algorithm, in brief, consists of two loops. The "inner loop" or *base learner* is a supervised learning algorithm. Specifically, [Yarowsky (1995)] uses a simple decision list learner that considers rules of the form, "If instance x contains feature f, then predict label j", and selects those rules whose precision on the training data is highest. The "outer loop" is given a seed set of rules to start with. In each iteration, it uses the current set of rules to assign labels to unlabeled data. It selects those instances on which the base learners predictions are most confident, and constructs a labeled training set from them. It then calls the inner loop to construct a new classifier (that is, a new set of rules), and the cycle repeats. The Algorithm presented in 3 can be considered as the conventional self-training algorithm.

5.2.3 Comparison between Self-training and Co-training

As discussed in [Abney (2004)], co-training [Blum & Mitchell (1998)] has subsequently become more popular, perhaps in part because it has proven amenable to theoretical analysis, in contrast to the Yarowsky algorithm, which is as yet mathematically poorly understood. The Yarowsky algorithm does have the advantage of placing less restriction on the data sets it can be applied to. Co-training requires data attributes to be separable into two views that are conditionally independent given the target label; the Yarowsky algorithm makes no such assumption about its data.

Both self-training and co-training are bootstrapping algorithms and their mission is to extend the original set of labeled examples. Under our opinion, the effect of a bootstrapping algorithm is depend on the quantity of added labeled examples which can be evaluated via two criteria: the accuracy of added examples; and how much of information will be achieved from new labeled examples. In our consideration, we regard the difference between self-training and co-training at the important point: self-training uses only one view while co-training using two views for representing data. The easy way to understand these views is that each view can be considered as a set of features. The view of self-training is the set containing all features and this set will split into two distinct sets which represent for the two views of co-training. Now, we discuss about these two criteria under the implementation of co-training and self-training algorithms.

- For the first criterion, it is natural that the more prior information the classifier contains, the more confidence of its detection is. Co-training algorithm uses two views with one of assumptions that each view its self is sufficient for detecting labels for new examples. But this assumption is impractical, and it is easy to see that even the set of all features is also lack of information for the labelling task. Therefore, for this criterion, self-training seems to more confident than co-training. Note that this problem concerns the problem P_2 as presented before.

- For the second criterion, at first we have to answer the question which information can be brought from new labeled examples? One kind of added information which is easy to see is information of new features. Under this consideration, if the assumptions of co-training is satisfied then it seems to be more useful than self-training because the two classifier trained on the two views will provide information for each other.

5.3 Proposed Solutions

Previously, we have identified three problems that may occur in semi-supervised learning methods and observed that overcoming them may effectively enhance the performance of learning algorithms in practical situations. In this section we will discuss solutions for these problems one by one, which form the basic for developing a new bootstrapping algorithm in the next section. From now on, we will distinguish two kinds of datasets: the added labeled dataset and the new labeled dataset. More particularly, in the semi-supervised learning algorithm, at each extension of the current labeled dataset, some new labeled examples will be added to the current dataset. Then, the set of such new labeled examples is called the added labeled dataset, and the union of the current labeled dataset with the added labeled dataset is called the new labeled dataset.

5.3.1 Imbalanced Data

```
Procedure 5 – Resize($L_0, L', \Delta$): Resizing class-based subsets
Input:
  $L_0$ is the original labeled dataset; $L'$ is the added labeled dataset
  $\Delta$ – tolerance in retaining class distribution
Output:
  $N' = \{n'_1, \ldots, n'_m\}$ – new sizes of class-based subsets for $L$

compute $N^0 = \{n^0_1, \ldots, n^0_m\}$ – sizes of class-based subsets of $L_0$.
compute $N' = \{n'_1, \ldots, n'_m\}$ – sizes of class-based subsets of $L'$.
compute $D^0 = \{d^0_1, \ldots, d^0_m\}$ – the class distribution of $L_0$
repeat
  set $N = \{n_1, \ldots, n_m\}$ by $n_i = n^0 + n'_i$, for $i = 1, \ldots, m$
  compute $D = \{d_1, \ldots, d_m\}$ from $N$
  if there exist $l$ such that $d_l - d^0_l > \Delta$ (*) then
    compute $r$ such that $\frac{n_l - r}{\sum_{i=1}^{m} n_i - r} = (d^o_l + \Delta)$
    if $n'_l > (r + 1)$ then
      $n'_l \leftarrow [n'_l - (r + 1)]$
    else
      $n'_l \leftarrow 0$
    end if
  end if
until condition (*) does not hold
```

Now we are concerned with situations in which class distribution of the original labeled dataset is biased, i.e. one or several classes considerably dominate the others. In such a case, adding new labeled examples in semi-supervised learning may make this bias stronger. This is because following the strategy of selecting new labeled examples based on classification probability, examples of dominant classes have more chance to be selected. This is clearly an undesired situation and has been considered in previous studies, e.g. [Blum & Mitchell (1998), Pierce & Cardie (2001)]. The solution for this problem is to retain the class distribution of the labeled dataset whenever it is extended. Furthermore, we are also based on the assumption that the class distribution of the original labeled dataset is similar the class distribution of the "real" data.

To tackle this problem, our solution is as follows. For a set of labeled examples, we divide examples into subsets such that all examples in each subset have the same label. We call these class-based subsets. From the new labeled examples which are obtained at the extension step, we must resize its class-based subsets such that the class distribution of the original labeled dataset can be retained. However, we may not always strictly retain this class distribution, such as in the case there are one or more class-based subsets which are empty. Therefore, developing a procedure for maintaining the class distribution should take this situation into account. In our case, we introduce a tolerance parameter by which we can avoid fixing the number of new labeled examples of each class is fixed, as in [Blum & Mitchell (1998), Pierce & Cardie (2001)]. The proposed solution for retaining class distribution is described in Procedure 5.

5.3.2 Increasing Confidence of New Labeled Data

Regarding this task, an usual approach is to use a supervised learning algorithm to train a classifier based on the labeled dataset, and then use this classifier to detect labels for the examples in a subset U' of the current unlabeled dataset U. Formally, let $\mathcal{L}$ be the set of labels (classes), and h be the supervised classifier. Given an example $\mathbf{e}$, the classifier h applied to $\mathbf{e}$ yields a probability distribution over $\mathcal{L}$, denoted by $P_h(\cdot|\mathbf{e})$. Then it is suggested that example $\mathbf{e}$ should be assigned to label $\hat{l}$ satisfying

$$\hat{l} = \arg\max_{l \in \mathcal{L}} P_h(l|\mathbf{e})$$

If $P_h(\hat{l}|\mathbf{e})$ is greater than a threshold α, then example $\mathbf{e}$ associated with label $\hat{l}$ will be added to L.

As mentioned previously, using a classifier with threshold α for determining new labeled examples may cause a tradeoff problem between the extendibility and the accuracy of label detection. Furthermore, an increase in threshold α does not always ensure an increase in accuracy of new labeled examples. Note that the extendibility of labeled data is not only depicted by the number of new examples added, but also by the "new information" brought by these added examples. Heuristically, a new example whose label is correctly detected with low confidence may bring richer and new information to the current labeled data, and therefore it may be useful to detect labels for new examples. Therefore, it would be helpful to find out such a way of extending labeled data which can maintain the extendibility while still ensuring the accuracy of labeled data. Here we also use a threshold-based method, but instead of designing a fixed value for the threshold, we design a set of values for the threshold such that at each iteration of the bootstrapping algorithm the best value is chosen based on estimating the upper bound of classification error of labeled data using the approach presented in [Goldman & Zhou, Zhou & Li (2005)]. Particularly, this selection of a threshold value is based on the evaluation of generated labeled datasets, which is done as follows.

Procedure 6 – DataEvaluate(L_0, L_{add}): Evaluation of Generated Labeled Data

Input: L_0 is the original labeled data; L_{add} is the added labeled data

```
1: $m \leftarrow |L_{add}|$
2: train on $L_{add}$ to generate classifier $h$
3: use $h$ to test on $L_0$ and obtain classification error rate $\eta$
4: return $q = m(1 - 2\eta)^2$
```

f a dataset has size m, this value has a relationship with the hypothesis accuracy $(1 - \varepsilon)$, where ε is the upper bound on the classification error rate of the hypothesis, in terms of the following formula which is presented in [Goldman & Zhou].

$$m = \frac{c}{\varepsilon^2(1-2\eta)^2} \text{ or, equivalently } \varepsilon = \sqrt{\frac{c}{m(1-2\eta)^2}} \tag{5.1}$$

where c is a constant, which is simply set to 1, and η is classification noise rate of the training data (η must be less than 0.5). Note that [Goldman & Zhou] generated this formula by simply using a theorem about Probably Approximately Correct (PAC) property in [Angluin & Laird(1988)]. More detail, the formula (5.1) is derived from the inequality (5.2), with the assumption that other parameters are held constant, as follows,

$$m \geq \frac{c}{\varepsilon^2(1-2\eta)^2} ln(\frac{2N}{\delta}) \tag{5.2}$$

where N is the number of hypothesis, and δ is the confidence, then a hypothesis H_i that minimizes disagreement with the training data will have PAC property:

$$Pr[d(H_i, H*) \geq \varepsilon] \leq \delta$$

where $d(,)$ is sum over the probability of elements from the symmetric different between the two hypothesis sets H_i and H^* (the ground-truth).

Assume that we are standing at the iteration t^{th}, denote by L^t the current labeled dataset, L^t_{add} is the current added labeled dataset, and w^t is the number of examples in $L^t \cup L^t_{add}$ that are mislabeled. Then $1/\varepsilon^2$ denoted by q^t can be estimated by

$$q^t = |L^t \cup L^t_{add}| \left(1 - 2\frac{2w^t}{|L^t \cup L^t_{add}|}\right)^2$$

The semi-learning process is continued if $q^t > q^{t-1}$, and the labeled dataset is updated: $L^{t+1} \leftarrow L^t \cup L^t_{add}$.

Here we also accept this approach for evaluating the generated labeled datasets (i.e. the generated classifiers), but with a difference regarding the estimation of experimental classification error rate η. To estimate η, [Goldman & Zhou] used a 10-fold test on $L^t \cup L^t_{add}$, while [Zhou & Li (2005)] also trained on the whole data (include original labeled data and new labeled data) but just tested on the original labeled data.

Also agree with the observation in [Zhou & Li (2005)] that it is difficult to estimate the classification error on the unlabeled examples, we therefore compute classification error rate by testing on only the original labeled examples, with a heuristic assumption that the unlabeled examples hold the same distribution as the labeled ones. Moreover, we use only the total number of labeled examples added through iterations without the original labeled examples for training while [Goldman & Zhou] and [Zhou & Li (2005)] used both these kinds of labeled data. The reason for our choice stems from the following observations:

- WSD always has the problem of over-fitting on training data, which means if we train and test on the same data, we often receive a very high accuracy (approximate 100%).
- It is natural that if the added labeled examples are correctly classified, the classifier trained on them will give high accuracy when testing on the original labeled examples.

The flexible and dynamic strategy for selecting values for the threshold α with the help of Procedure **DataEvaluate** is used for the task of extending labeled data and this procedure is sketched in Procedure 7.

Procedure 7 – **Extendibility**$(L, U, \Omega, \mathbf{A})$: Extend labeled data

```
Input:
  $L$ – the current labeled dataset
  $U$ – the current unlabeled data
  $\Omega = \{\alpha_i\}_{i=1}^{n}$ – a set of threshold values of size $n$
  $\mathbf{A} = \{A_1, \ldots, A_R\}$ is the set of supervised algorithms, which is used in the case of using multi–classifiers
set a pool of empty datasets, $\mathbf{L} = \{L_i = \emptyset\}_{i=1}^{n}$
get classifier $h$ trained on $L$
for all example $\mathbf{e} \in U$ do
    use $h$ to detect labels for $\mathbf{e}$, obtain a new labeled example $\mathbf{e}'$ with a overall corresponding support degree $P(e')$;
    for all $\alpha_i \in \Omega$ do
        if $P(e') > \alpha_i$ then
            add $\mathbf{e}'$ to $L_i$
        end if
    end for
end for
call Resize on $L_i$ and remove from $L_i$ a certain number of examples such that it is appropriate to the new size, for $i = 1, \ldots, n$.
evaluate $L_i$ by calling the procedure DataEvaluate, for $i = 1, \ldots, n$,
    and get the best $L_k$, where $k = \arg\max_{i} DataEvaluate(L, L_i)$
return $L_k$
```

Also regarding the problem P_2, we propose a solution motivated from the observation that combining classifiers has significantly improved the performance of supervised learning systems. Especially in the context of semi-supervised learning, where the relatively small amount of available labeled data would not be enough to build good classifiers, combination of different classifiers may hopefully enhance the quality of new labeled data by integrating complementary information extracted from individual classifiers about patterns to be classified. If we use classifier combination for detecting labels for unlabeled examples in the procedure of Extendibility, it is then called the use of multi-classifier. Otherwise, if we use only one classifier then it is called the use of single-classifier. Note that the use of either single classifier or multi-classifier is stated at Step 2 and Step 4 of Procedure 7. In the case of multi-classifier, we use a combination rule on a set of R classifiers to generate the classifier h. In such a case, R different classifiers correspond to R different machine learning algorithms taken from $\mathbf{A}$ trained on the current labeled data. For each unlabeled example $\mathbf{e}$, these R classifiers are combined to output overall support degrees associated with corresponding labels in $\mathcal{L}$. Then the highest overall support degree is used in comparison with the threshold in such a way that only the new labeled examples which have the highest overall support degrees greater than the threshold α are added to the current labeled data.

5.3.3 Generating the Final Classifier

Now we will discuss about problem P_3 of how to generate the final classifier. Regarding this problem, there are two issues to be addressed: when should we stop the process of enlarging the labeled dataset, and how to build the final classifier. In previous studies related to WSD as in [Pham *et al.* (2005), Mihalcea (2004)], the authors first design a development dataset and then run the semi-learning algorithm on this dataset several times to select a value for the iteration number which is used for test datasets. After that the classifier built on the labeled data obtained at the last iteration is chosen as the final one.

However, in our opinion, the optimized value of iteration number depends on each particular dataset, as well as depends on each polysemous word. Therefore, it would be better if this value can be dynamically determined by evaluating generated labeled datasets. For this purpose, we can use the evaluation method described previously in Procedure 6. In addition, as discussed above, we can also use techniques of classifier combination for building the final classifier. The details are shown below.

We first observe that during the process of extending labeled data in semi-supervised learning systems for WSD, the following two situations may happen: the first one is the feature space may be also expanded concurrently, due to some new features covered by new examples which have not been occurred in available examples previously; the second one is there may be some misclassified examples which were added to the labeled dataset at some steps of extending labeled data. Both these situations may lead to the following consequences: the generated classifiers may not much improve, or may even decrease the labelling quality for test examples which could have been correctly labeled by the initial supervised classifier; the generated classifiers may be better in detecting labels for test examples which contain many new features covered by new added labeled examples.

These observations suggest that the use of only one classifier built on the last labeled dataset as the final classifier may not always be the best solution. Again, we aim to apply strategies of combining classifiers for enhancing the labelling quality of the final classifier in semi-supervised learning for WSD. To this end, after each extension of labeled data we build the corresponding classifier and then combine all of them according to a combination strategy to obtain the final classifier. Note that, there is another alternative in which we just combine the classifiers which are trained on the original and the last labeled dataset.

5.3.4 A New Algorithm

Algorithm 8 – A New Bootstrapping Algorithm

Input:L_0 is the original dataset; $\mathbf{A} = \{A_1, \dots, A_R\}$ is the set of supervised algorithms; A^* is the primary supervised algorithm; C is a combination rule; $\mathbf{L}$ is the set of obtained labeled datasets; Δ is the tolerance which is used for retaining class distribution ; M is maximum number of unlabeled examples used at each iteration; Ω is a set of the threshold α.

Output: H – the final classifier

```
$k \leftarrow 0$; $q_0 \leftarrow 0$; $\mathbf{L} \leftarrow \{L_0\}$
repeat
    $k \leftarrow k + 1$;
    randomly get $M$ examples from $U$ to obtain $U' \subset U$
    $L' \leftarrow$ Extendibility$(L_k, U', \Omega, \mathbf{A})$
    $L_{k+1} \leftarrow L_k \cup L'$;
    $q_{k+1} \leftarrow$ DataEvaluate$(L_0, L_{k+1})$
    if $q_{k+1} > q_k$ then
        $\mathbf{L} \leftarrow \mathbf{L} \cup \{L_{k+1}\}$;
    end if
until $q_{k+1} < q_k$ or $L' = \emptyset$
train $A^*$ on $\mathbf{L}$ to generate a set of classifiers, $\mathbf{h} = \{h_0, \dots, h_t\}$
apply the combination rule, $C$, on $\mathbf{h}$ to generate the final classifier $H$
```

On the basis of the above discussions, we develop a new bootstrapping algorithm as shown in Algorithm 8. In this algorithm, at each iteration first M examples are randomly extracted from the whole unlabeled dataset U, we denote by U' the set of these unlabeled examples (this is necessary in the case that U is very large). After U' is selected, the procedure **Extendibility** is called to enlarge the current labeled dataset L_k (k is the current iteration number). In this step, an added labeled dataset, L', is generated, which is then combined with the current labeled dataset to yield a new labeled dataset, L_{k+1}. Note that when **Extendibility** has been carried out, procedure **Resize** is called to retain class distribution for the new labeled dataset. After obtaining L_{k+1}, it is evaluated by procedure **DataEvaluate**. This process is repeated until there are no more new labeled examples to be discovered, or the new labeled examples do not improve the labeled dataset. When this process stops, we obtain a set of new labeled datasets, namely $\mathbf{L} = \{L_0, \dots, L_t\}$ (note that, here $t = k$ or $t = k + 1$ depends on the conditions for stopping the loop of the algorithm, $q_{k+1} < q_k$ or $L' = \emptyset$). Then, we use the supervised learning algorithm A^* trained on $\mathbf{L}$ to obtain a set of different classifiers $\mathbf{h} = \{h_0, \dots, h_t\}$. Finally, the final classifier H is generated by applying the combination rule C on $\mathbf{h}$.

Note that, at the final step of this algorithm we can also apply the combination rule for only the classifier trained on the original labeled dataset and the classifier trained on the last new labeled dataset (i.e. L_0 and L_N). This is suggested by the observation that intermediately generated classifiers participating in the combination may decrease advantages of the last classifier. Further, using only the initial and the last classifiers in combination is also due to advantages in terms of time computation and storage space.

5.4 Experiment for Semi-Supervised Learning

5.4.1 Experimental Models

Actually, the proposed semi-supervised learning algorithm is the result of integrating solutions for problems P_1, P_2, and P_3 into the general bootstrapping algorithm. Therefore, to see how effective each of the proposed solutions or their combinations is, in the sequence we develop several different experimental models of the proposed semi-supervised learning algorithm.

As in Procedure **Extendibility**, we use a set of values for α instead of a fixed value. In particular, we define this set as $\Omega = \{0.5, 0.6, 0.7, 0.8, 0.9\}$. The upper bound ($\alpha = 0.9$) and lower bound ($\alpha = 0.5$) of these values are used for those models which follow the conventional threshold-based method, which is based on a fixed threshold. Particularly, the experimental models are as follows.

- Call the general bootstrapping algorithm M_0, without any proposed solutions of P_1, P_2, and P_3. In this model, we investigate two cases: $\alpha = 0.9$ and $\alpha = 0.5$.
- To investigate problem P_1, we design the model called M_1, which is the model M_0 plus the procedure *Resize*, i.e the solution for P_1. In this model, we also investigate two cases: $\alpha = 0.9$ and $\alpha = 0.5$
- The following models are designed to test the solution of P_2 in combination with the solution of P_1, with and without using a strategy of classifier combination. Here, the set of threshold values is $\Omega = \{0.5, 0.6, 0.7, 0.8, 0.9\}$.

  - $M_2^{flexible+single}$ is the model in which we use a flexible and dynamic selection over all values in Ω for α. Moreover, this model just uses a single classifier, namely the NB classifier, to detect labels for unlabeled examples.

  - $M_2^{flexible+combined}$ is similar to $M_2^{flexible+single}$ but instead of using one classifier, here we use three classifiers including NB, MEM, and SVM, and the median rule.

  Note that all these models use procedure **DataEvaluation** as the condition for stopping the loop of the algorithm.

- Regarding problem P_3, we design two experimental models as follows: in model M_3^{two} we just combine the initial classifier and the last classifier; and in model M_3^{all} we combine initial classifier and all generated classifiers.

These models are intuitively summarized in Table 5.1

5.4.2 Parameter Setting

Feature Selection

Features used in experiments for semi-supervised learning are determined as denoted in Chapter 3. Particularly, the set of overall features includes all features from F_1^a, F_1^b, F_1^c, F_2, F_3, F_5, F_6. This feature set is used for self-training based algorithms.

For co-training based algorithms, we have to design two views from the overall feature space, i.e. two distinguish feature sets. Like [Pham *et al.* (2005), Mihalcea (2004)] we

Table 5.1: Experimental Models of Bootstrapping

Model	P_1 olution	P_2 olution	P_3 solution
M_0, $\alpha = 0.9$			
M_0, $\alpha = 0.5$			
M_1, $\alpha = 0.9$	x		
M_1, $\alpha = 0.5$	x		
$M_2^{flexible+single}$	x	x (single classifier)	
$M_2^{flexible+combined}$	x	x (multiple classifiers)	
M_3^{two}	x	x	x (two classifiers)
M_3^{all}	x	x	x (all classifiers)

design two views such that one represents for local context and the other represents for topical context. For this purpose and based on the characteristics of each kinds of feature subsets in F_1^a, F_1^b, F_1^c, F_2, F_3, F_5, F_6, we designed two subsets as: $\{F_1^a, F_1^b, F_1^c\}$ for the first view, and $\{F_2, F_3, F_5, F_6\}$ for the second view.

Supervised Learning Algorithms

Naive Bayes (NB), MEM (Maximum Entropy Model), and SVM (Support Vector Machine) are chosen as supervised learning algorithms for procedure **Extendibility** in the case that a combination strategy is integrated in the solution of P_2. Otherwise, in the case of using single classifier instead of combining multiple classifiers we will use the NB classifier. Further, the NB algorithm is also used for A^* in Algorithm 8.

Combination Rule for Problem P_2

The combination rule $\Re$, which is used in the procedure *Extendibility* is required to output a probability distribution over the classes. Among the combination rules as mentioned in Chapter 5, we investigated max, min, and median/average rules on the datasets of the four words (including *interest*, *line*, *hard*, and *serve*) and the obtained result shows that the median/average rule gives the highest accuracy on the added labeled data (it also agrees with the experimental results as shown in Table. 4.3). Therefore, median rule is chosen as the combination rule $\Re$.

For the other parameters, we set $M = 500$ and $\Delta = 0.01$.

5.4.3 Results

The first test is for investigating the problem of imbalanced increasing of training data, i.e. P_1, with the experiment carried out on Senseval-2 and Senseval-3. The obtained results are shown in Fig. 5.6. In this experiment, we let the iteration run from 1 to 5, and compute the ratio of the largest class to the whole dataset (note that the result at iteration 0 corresponds to the original labeled data). As seen in the figure, the portion of the largest class (or dominated class) increases according to the increase of iteration. This reflects the imbalanced increasing of training data as discussed above.

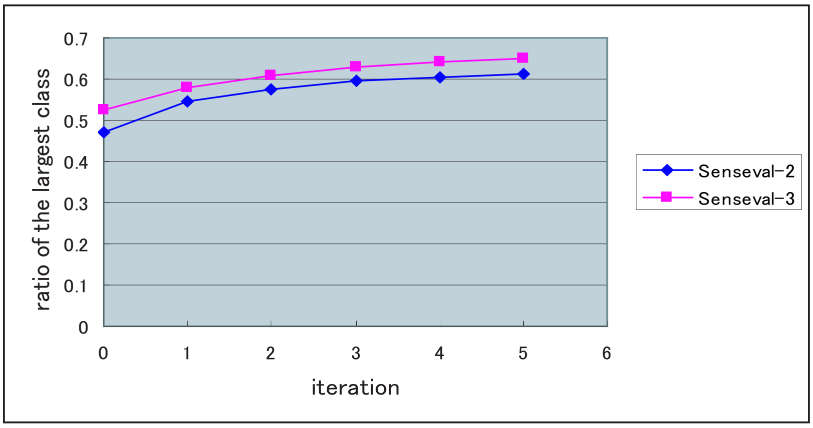


Figure 5.6: Test Problem P_1 on Senseval-2 and Senseval-3

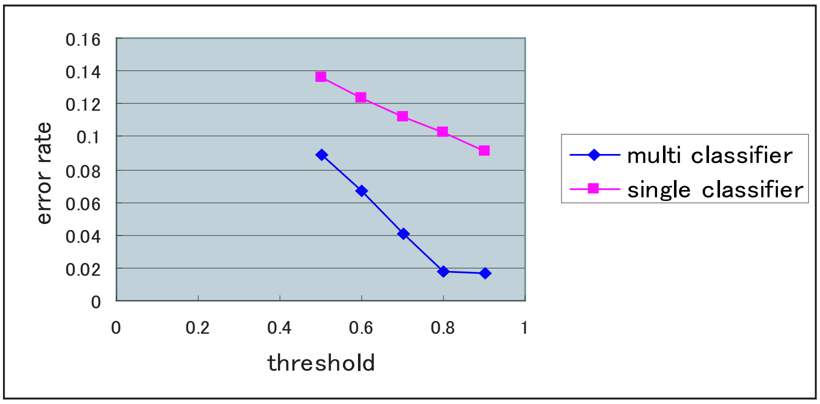


Figure 5.7: Test Problem P_2 with Self-Training

The second test is for investigating the problem of extending labeled data (problem P_2). For this purpose, we tested the algorithm on the datasets of four words including *interest*, *line*, *hard*, and *serve*. All examples in these datasets were tagged with the right senses. The sizes of these datasets are 2369, 4143, 4378, and 4342, respectively. These datasets are large enough for dividing into labeled and unlabeled datasets. We randomly select 100 examples for labeled data, 200 examples for test data, and the remaining examples are treated as unlabeled examples. Note that, because we knew the tagged senses of examples in unlabeled datasets, we are able to evaluate the correctness of the new labeled examples (for problem P_2). Fig. 5.7 and Fig. 5.8 show experimental results of the test using self-training and co-training respectively, in which two solutions corresponding to using a single classifier or multiple classifiers were investigated. As we can see in these figures, using multiple classifiers in combination yields a lower classification error rate. Further, Fig. 5.9 is the integration of these two figures, that shows a comparison between self-training and co-training concerning problem P_2. It shows that using self-training with the solution of classifier combination (i.e. multiple classifiers) gives the highest confidence (lowest classification error rate) of the new labeled examples.

Table 5.2 shows the results for the test of P_1 and P_2 problems, in which the conventional

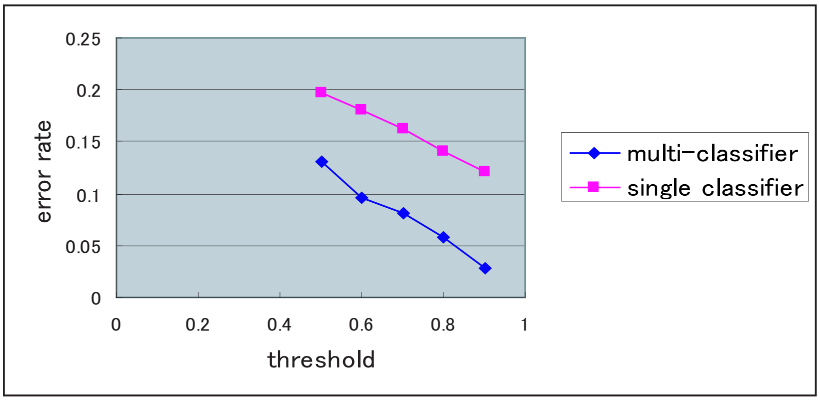


Figure 5.8: Test Problem P_2 with Co-Training

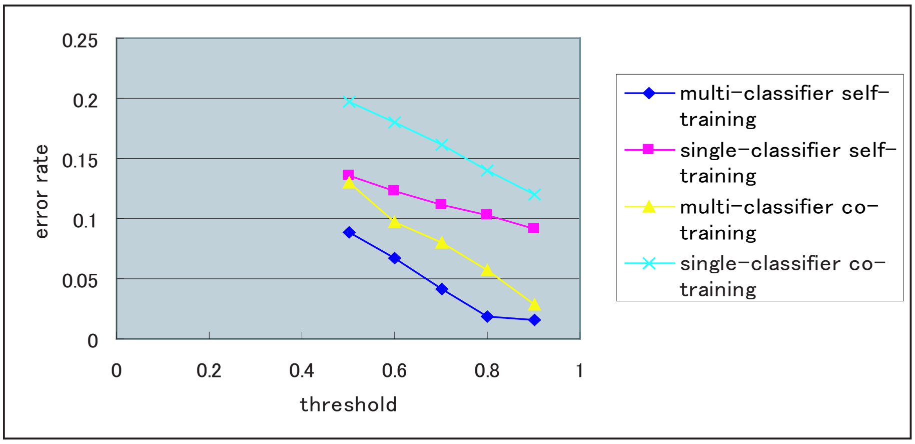


Figure 5.9: Test Problem P_2, Integrating Two Graphs: Self-Training and Co-Training

Table 5.2: Test Problem P_1 and P_2

		Self-Training		Co-Training	
	parameter	Senseval-2	Senseval-3	Senseval-2	Senseval-3
Supervised (NB)		64.05	71.96	64.05	71.96
M_0	$\alpha = 0.9$	62.07	71.70	61.61	68.79
M_0	$\alpha = 0.5$	62.97	71.07	58.93	66.61
M_1	$\alpha = 0.9$	63.89	71.65	62.81	70.74
M_1	$\alpha = 0.5$	64.19	71.65	61.24	68.99
$M_2^{flexible+single}$	Ω	64.63	72.44	62.42	68.99
$M_2^{flexible+combine}$	Ω	**65.70**	**72.64**	64.49	70.99

Table 5.3: Test Problem P_3: Results on Senseval-2 and Senseval-3

		DS1	NB	max	min	med	FM1	FM2	mvote	wvote
Self-Training										
Sen-2	M_3^{two}	66.23	66.07	66.27	**66.30**	66.25	66.27	**66.30**	65.35	66.27
	M_3^{all}	66.11	65.83	66.04	**66.27**	65.74	65.70	65.72	65.70	65.63
Sen-3	M_3^{two}	73.25	73.10	73.27	73.23	**73.30**	73.28	73.23	72.49	73.28
	M_3^{all}	73.20	73.12	73.22	72.74	**73.33**	**73.33**	**73.33**	73.20	**73.35**

Table 5.4: A comparison between Supervised Learning and Semi-Supervised Learning on Senseval-2 and Senseval-3

	Supervised Learning			M_3^{two}	
	NB	SVM	MEM	*max* rule	best rule
Senseval-2	64.05	63.72	64.79	**66.27**	**66.30**
Senseval-3	71.96	70.87	71.91	**73.27**	**73.30**

self-training algorithm, denoted by M_0, and the models M_1 and M_2 were implemented, where NB classifier is used as the baseline. From these results, we have the following conclusions.

- Better results given by model M_1 in comparison with model M_0 reflect that using the procedure of retaining class distribution is effective.
- Models M_2 give better results in comparison with models M_1. This shows that the proposed solutions for problem P_2 are quite effective. In addition, using flexible determination of α integrated with a strategy of classifier combination gives the best result.
- Only model M_2 yields better results in comparison with baseline results. With the proposed solutions for P_1 and P_2, we have shown that unlabeled data can significantly improve the performance of supervised learning.

Table 5.3 shows the results for the test of P_3. As we have seen, the results from these two models M_3^{two} and M_3^{all} are not much different. Therefore, the model M_3^{two} should be chosen due to the cost of time computation and storing space (M_3^{two} just combine the initial classifiers and the last classifier, while M_3^{all} combine all classifiers generated at each extension of labeled data). Further, we also see that the max rule used for the combination of generated classifiers gives acceptable results which approximately reach the best results in most cases.

In summary, Table 5.4 shows a comparison between the supervised learning algorithms and the selected semi-supervised learning model, namely M_3^{two}. For supervised learning, we implemented three algorithms including NB, MEM, and SVM. The obtained results show that model M_3^{two} yields better results in comparison with supervised WSD.

In Fig. 5.10, we have shown the effects of proposed solutions for problems P_1, P_2, and P_3. In the models $M0$, $M1$, and $M2$, we accept $\alpha = 0.9$, and for model $M3$ we

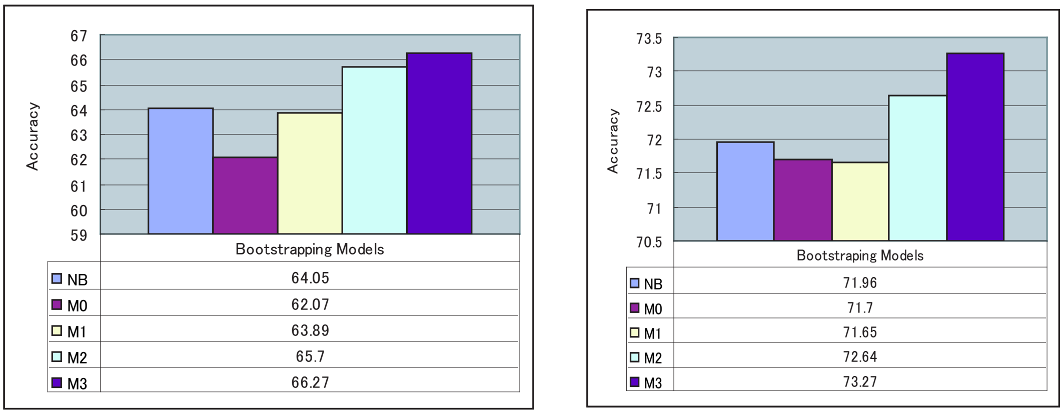


Figure 5.10: A comparison between bootstrapping models on Senseval-2 and Senseval-3

use the combination between the initial and the last classifiers by the max rule. This figure gives us a clearer view of effectiveness of proposed solutions as mentioned above: M_1 is better M_0(except the case of test on Senseval-3, but with a slight decrease); M_2 is better M_1 ; and M_3 is better M_2; Further, only with these solutions, the bootstrapping algorithm (models M_2 and M_3) gives better results in comparison with supervised learning. Note that with the conventional bootstrapping algorithm (model M_0), we receive a lower accuracy in comparison with supervised learning. It is also worth to emphasize that in [Le *et al.* (2006b)], we have shown that without using the solution for problem P_1 in models M_2 and M_3 we will receive a lower accuracy.

5.5 Combination for Post Data Extension

In Chapter 4 we have shown that classifier combination strategies is quite useful for improving performance of individual classifiers. In this chapter, unlabeled data is also shown to be useful for improving supervised learning. The problem here is how to combine advantages from using classifier combination techniques and using unlabeled data into a consensus strategy. To this end, a natural way is to apply classifier combination techniques on both kinds of individual classifiers, one is based on the original training data and the other is based on the extended training data which is obtained from semi-supervised learning. Here, for this task we propose a combination model which is designed based on the two following selections. We call this model the combination model for post-data-extension or the combination model for post semi-supervised learning.

– As shown in Chapter 4, there are two types of individual classifiers used in combination strategies, one is based on different context representations and the other is based on different supervised learning algorithms. Among them, the approach in which the individual classifiers are generated by using different supervised learning algorithms trained on the labeled data, give the best result. Therefore, we will follow this approach.

– In a combination model for post data extension, we first have to design individual classifiers. Assume that we have R different supervised learning algorithms, then R first individual classifiers are generated by using these algorithms trained on the original labeled data. To utilize the result of exploiting data we can also use these R algorithms

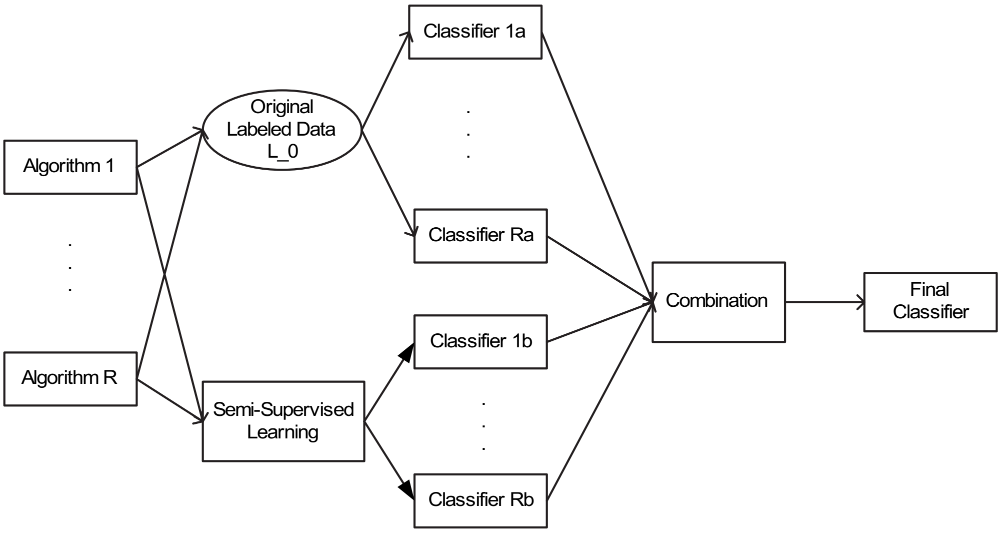


Figure 5.11: Combination Strategy for Post Labeled Data Extension

Table 5.5: Classifier Combination at Post Semi-supervised Learning

	DS	NB	max	min	med	FM1	FM2	mvote	wvote	meta-vote
Sen-2	67.1	66.8	66.8	66.8	67.0	67.1	**67.5**	66.4	66.9	**67.3**
Sen-3	73.7	73.8	73.4	73.6	**74.0**	73.9	**74.0**	73.7	**74.1**	**74.0**

trained on the extended labeled data (last labeled datasets). However, as shown in Algorithm 8, the extended labeled data is associated with an algorithm (denoted by A^*) which is used to evaluate this data. Therefore, it suggests us to use each algorithms in these R algorithms as the algorithm A^* and consequently generate R extended labeled datasets. And then using these R extended labeled datasets to generate more R individual classifiers. Now, we have $2R$ individual classifiers and they are combined as in Fig. 5.11.

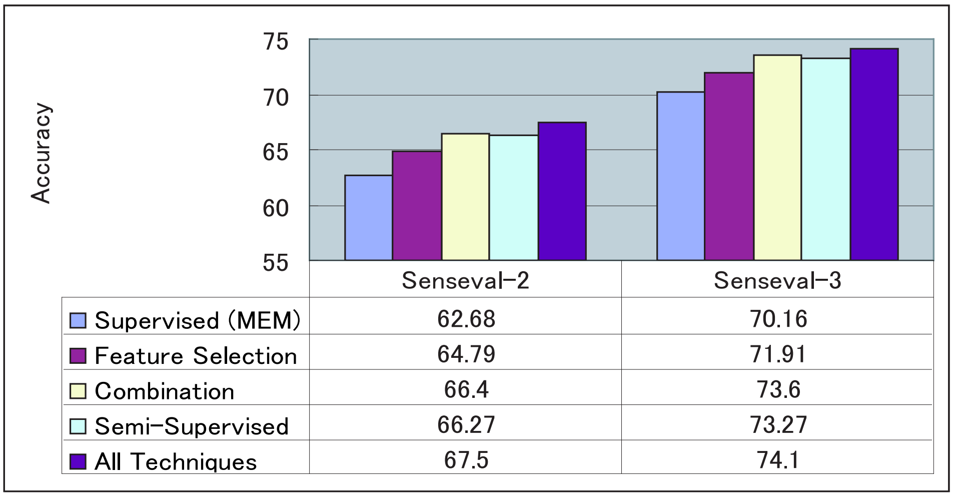
Experiment for Post Semi-Supervised Learning

Fig. 5.5 shows the experimental result of the proposed combination model for Post Semi-Supervised Learning. In this experiment, we also use the three supervised learning algorithms including NB, SVM, and MEM to generate 6 different individual classifiers for the the combination model, in which 3 classifiers among them are generated by training NB, SVM, and MEM on the original labeled data. For generating the 3 remain classifiers, we first run the proposed semi-supervised learning three times corresponding to three options of the algorithm $A*$, including NB, SVM, and MEM. Then, we obtain the classifier h_N at each of the three runs which is used as a remain classifiers (see Algorithm 8 for the detail). Actually, the 3 remain classifiers are generated by training NB, SVM, and MEM on the final extended labeled data obtained from the three runs of Algorithm 8 above, respectively.

Comparing result in Fig. 5.5 with result in Fig. 5.4, we can see that the combination

Table 5.6: A Comparison with Current WSD Systems

Previous Study	Methods	Senseval-2	Senseval-3
Our system	FS + Combination + Semi Baseline (MEM)	**67.5** 62.68	**74.1** 70.16
[Ando(2006)]	ASO multi-task + Semi Baseline (ASO single-task)	(*68.1*) 65.3	**74.1** 73.0
[Florian & Yarowsky (2002)]	Classifier Combination	66.5	
[Wu *et al.* (2004)]	Polynomial KPCA	65.8	
[Lee & Ng(2002)]	SVM	65.4	
The best systems in Senseval contests		64.2	72.9



	Senseval–2	Senseval–3
Supervised (MEM)	62.68	70.16
Feature Selection	64.79	71.91
Combination	66.4	73.6
Semi–Supervised	66.27	73.27
All Techniques	67.5	74.1

Figure 5.12: Summary of Proposed Methods

model for the purpose of combining advantages from classifier combination and semi-supervised learning is more effective in improving accuracy than the combination model which just focus on utilizing advantages from the original labeled data and the extended labeled data (as mentioned in problem P_3). Further, when comparing with result of the semi-supervised learning model without applying combination techniques as shown in Table 5.2 (i.e. models M_1 and $M_2^{flexible+single}$), we see that result of the combination model for Post Semi-Supervised Learning is much better.

Figure 5.12 summarizes the results of all proposed methods. From this figure, we can see that all proposed method, including "Feature Selection", "Combination", and "Semi-Supervised" improve accuracy of a supervised learning method (using the Maximum Entropy Model). Note that in "Combination" and "Semi-Supervised" we have already used the result of "Feature Selection". Furthermore, by combining all these methods, we archive better results in comparison with using separately each of them.

In summary, the proposed combination model for Post Semi-Supervised Learning is effective in combining advantages from both models of classifier combination and semi-supervised learning. The obtained result reaches the state-of-the-art WSD systems, as shown in Table 5.6, where: "FS" stands for "Feature Selection"; "ASO" stands for "Alternating Structure Optimization"; and MEM is simply "Maximum Entropy Model", SVM is "Support Vector Machines". Note that, in [Ando(2006)], for the method of "ASO multi-task+Semi", Senseval-2 data is used as the development data to find the optimized parameters for the test on Senseval-3 data. Therefore, for the test on Senseval-2 data, our result is the best, and for the test on Senseval-3 data, our result is the same [Ando(2006)] and is also the best.

5.6 Summary

This chapter presents one of major contents in our thesis – semi-supervised learning for WSD. We have identified three problems that may occur in semi-supervised learning methods and particularly investigated them for WSD. We have proposed solutions for these problems which form the basic for developing a new bootstrapping algorithm. To test the effective of the proposed solutions, we have generated various models of the new bootstrapping algorithm and tested them on Senseval-2 and Senseval-3. The experimental results show that the proposed solutions are effective for improving semi-supervised learning for WSD, and unlabeled data can significantly improve supervised WSD as well. Furthermore, we also developed a combination model for post semi-supervised learning with the purpose of combining advantages from classifier combination and semi-supervised learning, and with this model we have reached to the state-of-the-art WSD system.

Particularly, we have made the contribution as follows.

- A new bootstrapping algorithm with several variants are generated, with which unlabeled data is shown effective in improving supervised WSD while this improvement does not appear with the conventional bootstrapping algorithms.
- A novel combination model for post semi-supervised learning is proposed, which aims to combine advantages from classifier combination and semi-supervised learning. Experimental result of this model has reached the state-of-the-art of WSD systems.
- This chapter has also made a comparison between self-training and co-training respect to the confidence of new labeled examples. Some discussion was presented and experimental results showed that self-training give better result than co-training when applying for WSD problem.

Chapter 6

Conclusion and Future Directions

6.1 Summary of the Thesis

In this thesis, we have presented a study on the problem of Word Sense Disambiguation with the purpose of improving quality of the task by focusing on feature selection (investigating various kinds of knowledge sources to represent context, and applying some feature selection methods) and two "hot" aspects of machine learning approaches including classifier combination and semi-supervised learning. The study considers the research problems both on theoretical and practical views that make WSD more accuracy (reach to the state-of-the-art), and develop machine learning which can be applied for not only WSD problem but also other classification problems. The thesis consists of six chapters. The first chapter presents an overview of the thesis. The second chapter first presents our survey on the approaches in previous studies, and then presents the three supervised learning algorithms (Naive Bayes, Support Vector Machines, and Maximum Entropy Models) which are used as the basic algorithms in proposed methods. The next three chapters deal with three main tasks of thesis, respectively: context representation and feature selection; classifier combination; and exploiting unlabeled data. Chapter 6 contains the summary of the thesis and future research directions.

The motivations and proposed solutions for the problems mentioned in this thesis are based on investigating and solving the limitations of various related WSD studies. Experimental results were conducted on standard datasets (Senseval-2 and Senseval-3) and were compared to state-of-the-art systems in the field. All in all, the major contributions of the thesis are summarized as follows:

- The first contribution comes from the work on context representation and feature selection. We have proposed a machine learning approach to this word. The proposed method for knowledge sources determination is based on the simultaneously use of the Forward Sequential Selection and Backward Sequential Selection algorithms. As the result, the useful knowledge sources which represents context were determined. After obtaining the selected knowledge sources, we applied a filter method with using two feature measures, frequency and information-gain, to select useful individual features. The experimental result suggests that we should use the whole features from the selected knowledge sources. This work has provided an reasonable explanation for the use of selected features. In addition, we also proposed the simultaneously use of topical context in three window sizes (small, medium, and large), and this gives higher accuracy in comparison with using only the large size as in previous studies.

- The second contribution comes from the work on combining classifiers for WSD. In this work, two new approaches of classifier combination for WSD, which are based on Dempster-Shafer theory of evidence and OWA operators have been presented. Various combination rules were derived. The second-layer combination strategies were proposed including meta combination and meta-stacking. Various tests corresponding to various combination models were conducted and the obtained results show that meta-voting in meta combination strategies gives the best result and much improves accuracy of individual classifiers.

- The third contribution comes from the work on semi-supervised learning for WSD, that helps reduce the need of labeled training data by gaining additional information from a large amount of unlabeled data. We followed the approach in which the original labeled data is iteratively extended from unlabeled data. Two particular bootstrapping algorithms investigated are self-training and co-training. We first identify problems which may occurring in this approach and then proposed solutions for them. As the result, a new bootstrapping algorithm with several variants are generated. With the new algorithm, unlabeled data is shown effective in improving WSD. Furthermore, a novel combination model for post semi-supervised learning has been proposed, which aims to combine advantages from classifier combination and semi-supervised learning. Experimental result of this model reaches the state-of-the-art of WSD systems.

6.2 Future Directions

In the future work, we will focus on the remaining issues in this thesis. We also aim to build a real WSD system which is still a difficult and open problem in this field. Beside that we also want to continue developing the machine learning methods which have been presented in this thesis. Toward these purposes, we will concentrate on the following issues:

- In this thesis, a machine-learning-based method for knowledge determination and feature selection was proposed and an investigation of various kinds of knowledge was presented in Chapter 3. There are still some kinds of knowledge which have not been investigated in this work such as thesauri, sense definition in dictionary. For the future work, we will focus on utilizing the information contained in dictionary and then combined with results obtained in this thesis.

- We will continue developing machine learning algorithms specially in the field of semi-supervised learning. In Chapter 5, we proposed a new bootstrapping algorithm applied for the approach in which the original labeled data is iteratively extended. In our opinion the proposed solutions, which help to generate the new bootstrapping algorithm, can be also applied for other semi-supervised learning approaches, such as Transductive Support Vector Machines, Graph-based algorithms. Concerning the problem of applying classifier combination techniques for WSD, we will investigate one more one aspect of machine learning to determine weights for individual classifiers, that is the method based on minimizing error rate training.

- As mentioned in this thesis, every researches in WSD are still far from applying in practice, due to the low accuracy of these systems (just around 70%). Toward developing a WSD system which can be applied in practice, such as in machine translation systems, information retrieval, etc. we will focus on an aspect of machine learning, that is "Active

Learning". The results obtained in this thesis will be utilized in the strategy of Active Learning for two purposes: building useful corpora with as less as possible of human efforts; and improving accuracy of WSD systems.

Bibliography

[Angluin & Laird(1988)] D. Angluin, and P. Laird, 1988. Learning from noisy examples. *Machine Learning*, Vol. 2, pages 343–370.

[Al-Ani & Deriche 2002] A. Al-Ani and M. Deriche, A new technique for combining multiple classifiers using the Dempster-Shafer theory of evidence, *Journal of Artificial Intelligence Research* **17** (2002) 333–361.

[Abney (2004)] S. Abney, 2004. Understanding the Yarowsky Algorithm. *Computational Linguistics*, 1 September 2004, vol. 30, no. 3, pp. 365–395

[Ando(2006)] R. K. Ando, 2006. Applying Alternating Structure Optimization to Word Sense Disambiguation, *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X)*, pages 77–84.

[Agirre & Martinez (2001)] E. Agirre, and D. Martinez, 2001a. Decision lists for english and basque. *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL/EACL* Toulouse, France.

[Agirre & Rigau (1996)] E. Agirre and G. Rigau, 1996. Word Sense Disambiguation using Conceptual Density. *Proceedings of the 16th International Conference on Computational Linguistic (COLING)*, Copenhagen, Denmark.

[Blum & Chawla (2001)] A. Blum and S. Chawla, 2001. Earning from Labeled and Unlabeled Data Using Graph Mincuts. *Proceedings of ICML*, pages 19–26.

[Blum & Mitchell (1998)] A. Blum, and T. Mitchell, 1998. Combining labeled and unlabeled data with co-training. *Proceedings of COLT*, pages 92–100.

[Brill & Wu (1998)] E. Brill, J. Wu, 1998. Classifier Combination for Improved Lexical Disambiguation *Proceedings of COLING–ACL*, pages. 191–195.

[Berger *et al.* (1996)] A. Berger, S. Della Pietra, and V. Della Pietra, 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22:39–71.

[Brody *et al.* (2006)] S. Brody, R. Navigli, M. Lapata. Ensemble Methods for Unsupervised WSD, 2006. *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics joint with the 21st International Conference on Computational Linguistics (COLING-ACL 2006)*, Sydney, Australia, July 17–21st, 2006, pages. 97–104

[Brown *et al.* (1991)] P.F. Brown, S.A. Della Pietra, V.J. Della Pietra, and R.L. Mercer, 1991. Word Sense Disambiguation Using Statistical Methods. *Proceedings of ACL*, pages 264–270.

[Bruce & Wiebe (1994)] R. Bruce and J. Wiebe, 1994. Word Sense Disambiguation using Decomposable Models. *Proceedings of ACL*, pages 139–145.

[Charniak (2000)] E. Charniak. 2000. A maximum-entropy-inspired parser. *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 132–139.

[Cowie *et al.* (1992)] Cowie, Jim, Joe A. Guthrie, and Louise Guthrie. 1992. Lexical disambiguation using simulated annealing. *Proceedings of the 14th International Conference on Computational Linguistics, COLING'92*, volume 1, pages 359–365.

[Collins & Singer (1999)] M. Collins, Y. Singer, 1999. Unsupervised Models for Named Entity Classification. *Proceeding of EMNLP*, pages 100–111.

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

[Diab & Resnik (2002)] M. Diab and P. Resnik, 2002. An Unsupervised Method for Word Sense Tagging Using Parallel Corpora. *Proceedings of ACL*, pages 255–262.

[Dill *et al.* (2003)] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R.V. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J.A. Tomlin, and J.Y. Zien, SemTag and Seeker: Bootstrapping the semantic web via automated semanticannotation. *Proceedings of the Twelfth International World Wide Web Conference*, pages 178–186.

[Domingos (1997)] P. Domingos, 1997. Context-sensitive feature selection for lazy learners. *Artificial Intelligence Review*, (11):227-253.

[Escudero *et al.* (2000a)] G. Escudero, L. Marquez, and G. Rigau, 2000. Naive Bayes and exemplar-based approaches to Word Sense Disambiguation revisited. *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI)*, pages 421–425.

[Escudero *et al.* (2000b)] G. Escudero, L. Marquez, and G. Rigau, 2000. Boosting Applied to Word Sense Disambiguation. *Proceedings of the 11th European Conference on Machine Learning (ECML)*, pages 129–141.

[Frank *et al.* (2003)] E. Frank, M. Hall, and B. Pfahringer, 2003. Locally Weighted Naive Bayes. *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, pages 249–256.

[Fukumoto & Suzuki (1996)] F. Fukumoto and Y. Suzuki, 1996. An Automatic Clustering of Articles Using Dictionary Definitions. *Proceedings of the 16th International Conference on Computational Linguistics*, papges 406–411.

[Florian *et al.* (2002)] R. Florian, S. Cucerzan, C. Schafer, and D. Yarowsky, 2002. Combining Classifiers for Word Sense Disambiguation. *Journal of Natural Language Engineering*, Vol. 8(4) pages 327–341.

[Florian & Yarowsky (2002)] R. Florian and D. Yarowsky, 2002. Modeling consensus: Classifier combination for Word Sense Disambiguation. *Proceedings of EMNLP 2002*, pages 25–32.

[Gale *et al.* (1993)] Gale, William A., Kenneth W. Church, and David Yarowsky. 1993. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439.

[Goldman & Zhou] S. Goldman and Y. Zhou, 2000. Enhancing supervised learning with unlabeled data. *Proceedings of ICML*, pages 327–334.

[Hearst (1991)] M.A.Hearst, 1991. Noun homograph disambiguation using local context in large corpora. *Proceedings of the Seventh Annual Conference of the Centre for the New OED and Text Research: Using Corpora*, pages 1–22, Oxford, UK.

[Herrera & Verdegay (1996)] F. Herrera and J. L. Verdegay, 1996. A linguistic decision process in group decision making, *Group Decision Negotiation* (5):165–176.

[Hoste *et al.* (2002)] V. Hoste, I. Hendrickx, W. Daelemans, and A. van den Bosch, 2002. Parameter Optimization for Machine Learning of Word Sense Disambiguation. *Natural Language Engineering*, Vol. 8(3), pages 311–325.

[Ide *et al.* (1998)] N. Ide and J. Véronis, 1998. Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art. *Computational Linguistics* Vol. 24, pages 1–40.

[Joachims (1999)] T. Joachims, 1999. Transductive Inference for Text Classification Using Support Vector Machines. *Proceedings of ICML*, pages 200–209.

[Joachims (2003)] T. Joachims, 2003. Transductive Learning via Spectral Graph Partitioning. *Proceedings of ICML*, pages 290–297.

[John *et al.* (1994)] G.H. John, R. Kohavi, and K. Pfleger, 1994. Irrelevant Feature and the Subset Selection Problem. *Proceedings of ICML*, pages 121-129.

[Karov & Edelman (1998)] Y. Karov and S. Edelman, 1998. Similarity–Based Word Sense Disambiguation. *Computational Linguistics*, Vol. 24(1), pages 41–59.

[Klein *et al.* (2002)] D. Klein, K. Toutanova, H. Tolga Ilhan, S.D. Kamvar, and C.D. Manning, 2002. Combining Heterogeneous Classifiers for Word–Sense Disambiguation. *ACL WSD Workshop*, pages 74–80.

[Kittler *et al.* (1998)] J. Kitter, M. Hatef, R. Duin and J. Matas, 1998. On combining Classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 2(3), pages 226–239.

[Kira & Rendell (1992)] K. Kira and L. A. Rendell, 1992. The Feature Selection Problem: Traditional Methods and a New Algorithm. *Proceedings of AAAI*, pages 129–134.

[Kulkarni & Pedersen (2005)] A. Kulkarni, T. Pedersen: SenseClusters: Unsupervised Clustering and Labeling of Similar Contexts. *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 105–108.

[Krovets & Croft (1992)] R. Krovets and W.B. Croft, 1992. Lexical Ambiguity and Information Retrieval. *ACM Transaction on Information Systems*, Vol 10(2), pages 115–141.

[Krovets *et al.* (1997)] R. Krovets, 1997. Homonymy and Polysemy in Information Retrieval. *Proceedings of ACL/EACL97*, pages 72–79.

[Kilgarriff & Rosenzweig (2000)] A. Kilgarriff and J. Rosenzweig, 2000. Framework and Results for English SENSEVAL. *Computers and the Humanities.* Vol 36 pages 15–48.

[Kilgarriff (2000)] A. Kilgarriff, 2002. English Lexical Sample Task Description, *Proceedings of Senseval–2 Workshop, ACL2002*, pages 17–20.

[Kilgarriff (1997)] A. Kilgarriff, 1997. What is word sense disambiguation good for? *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pages 209–214.

[Kuncheva 2001] L. I. Kuncheva, 2001. Combining classifiers: Soft computing solutions, *Pattern Recognition: From Classical to Modern Approaches*, World Scientific, pages 427–451.

[Li & Li] H. Li and C. Li, 2004. Word Translation Disambiguation Using Bilingual Bootstrapping. *Computational Linguistics*, Vol. 30(1), pages 1–22.

[Lin (1997)] D.K. Lin, 1997. Using Syntactic Dependency as Local Context to Resolve Word Sense Ambiguity. *Proceedings of ACL*, pages 64–71.

[Le & Shimazu (2004)] C.A. Le and A. Shimazu, 2004. High Word Sense Disambiguation Using Naive Bayesian Classifier with Rich Features. *The 18th Pacific Asian Conference on Linguistic Information and Computation (PACLIC18)*, pages 105–113.

[Le *et al.* (2005a)] C.A. Le, V.N. Huynh, H.C Dam, A. Shimazu, 2005. Combining Classifiers Based on OWA Operators with an Application to Word Sense Disambiguation. *Proceedings of RSFDGrC*, Vol. 1, pages 512–521.

[Le *et al.* (2005b)] C.A. Le, V.N. Huynh, and A. Shimazu, 2005. Combining Classifiers with Multi-Representation of Context in Word Sense Disambiguation. *The 19th Pacific Asian Conference on Knowledge and Data Mining (PAKDD05)*, pages 262–268.

[Le *et al.* (2005c)] A.C. Le, V.N. Huynh, A. Shimazu, 2005. An Evidential Reasoning Approach to Weighted Combination of Classifiers for Word Sense Disambiguation. *The International Conference on Machine Learning and Data Mining (MLDM–2005)*, pages 516–525.

[Le *et al.* (2006a)] C. A. Le, A. Shimazu, and V.N. Huynh, 2006. Word Sense Disambiguation by Combining Classifiers with an Adaptive Selection of Context Representation. *Journal of Natural Languague Processing*, Vol 13, No. 1, pages 75–95.

[Le *et al.* (2006b)] C. A. Le, A. Shimazu, and M.L. Nguyen, 2006. Investigating Problems of Semi-Supervised Learning for Word Sense Disambiguation, *Proceedings 21st International Conference on Computer Processing of Oriental Languages (ICCPOL 2006)*, 12/2006, Singapore.

[Le *et al.* (2006c)] A.C. Le, V.N. Huynh, A. Shimazu, H.C. Dam, 2006. Weighted combination of classifiers for word sense disambiguation based on Dempster-Shafer theory. *The International Conference on Research, Innovation and Vision for the Future (RIVF–2006)*, pages 133–138.

[Le *et al.* (2006d)] A.C. Le, V.N Huynh, , A. Shimazu, and Y. Nakamori, 2006. Combining Classifiers for Word Sense Disambiguation Based on Dempster-Shafer Theory and OWA Operators. *Journal of Data and Knowledge Engineering (DKE)* (Accepted).

[Le *et al.* (2006e)] A.C. Le, A. Shimazu, V.N Huynh, and L.M. Nguyen, 2006. Semi-Supervised Learning Integrated with Classifier Combination for Word Sense Disambiguation. (Revising) *Journal of Computer Speech and Language.*

[Lee *et al.* (2004)] Y. K. Lee, H. T. Ng, and T. K. Chia, 2004. Supervised word sense disambiguation with support vector machines and multiple knowledge sources. *In Rada Mihalcea and Phil Edmonds, editors, Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 44–48.

[Lee & Ng(2002)] Y. K. Lee and H. T. Ng, 2002. An Empirical Evaluation of Knowledge Sources and Learning Algorithms for Word Sense Disambiguation. *Proceedings of EMNLP*, pages 41–48.

[Lesk (1986)] M. Lesk, 1986. Automated Word Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone. *Proceedings of the ACM SIGDOC Conference*, pages 24–26.

[Leacock (1998)] C. Leacock, M. Chodorow, and G. Miller, 1998. Using Corpus Statistics and WordNet Relations for Sense Identification. *Computational Linguistics*, pages 147–165.

[Lytinen (1986)] S.L. Lytinen, 1986. Dynamically combining syntax and semantics in natural language processing. *Proceedings of AAAI*, pages 574–578.

[Magnini *el al.* (2002)] Magnini, B., Strapparava, C., Pezzulo, G., & Gliozzo, A. (2002). The Role of Domain Information in Word Sense Disambiguation. *Natural Language Engineering*, 8(4): 359–373.

[Maruyama (1990)] H. Maruyama, 1990. Structural disambiguation with constraint propagation. *Proceedings of ACL–90*, pages 31–38.

[McCord (1990)] M. C. McCord, 1990. Slot Grammar: A system for simpler construction of practical natural language grammars. *Natural Language and Logic: International Scientific Symposium*, Lecture Notes in Computer Science, pages 118–145.

[Mooney (1996)] R.J. Mooney, 1996. Comparative Experiments on Disambiguating Word Senses: An Illustration of The Role of Bias in Machine Learning. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 82–91.

[Mihalcea & Moldovan (1999)] R. Mihalcea and D. Moldovan, 1999. A Method for word sense disambiguation of unrestricted text. *Proceedings of ACL*, pages 152–158.

[Mihalcea *et al.* (2004b)] R. Mihalcea , T. Chklovski, and A. Killgariff, 2004. The Senseval-3 English Lexical Sample Task, *Proceedings of ACL/SIGLEX Senseval-3*, pages 25–28.

[Mihalcea (2004)] R. Mihalcea, 2004. Co-training and Self-training for Word Sense Disambiguation, 2004. *In Proceedings of CoNLL*, pages 33–40.

[Mihalcea (2002)] Rada Mihalcea, 2002. Word Sense Disambiguation Using Pattern Learning and Automatic Feature Selection. *Journal of Natural Language and Engineering (JNLE)*, December 2002

[Montoyo *et al.* (2005)] A. Montoyo, A. Suarez, G. Rigau and M. Palomar, 2005. Combining knowledge and corpus-based Word-Sense-Disambiguation methods. *Journal of Artificial Intelligence Research*, 23: 299–330.

[McCarthy *et al.* (2004)] D. McCarthy, R. Koeling, J. Weeds, and J. Carroll, 2004. Finding Predominant Word Senses in Untagged Text. *Proceedings of ACL*, pages 279–286.

[Miller & Uyar (1997)] D.J. Miller and H.S. Uyar, 1997. A Mixture of Experts Classifier with Learning Based on Both Labelled and Unlabelled Data. *Advances in Neural Information Processing Systems*, pages 571–577.

[Nigam *et al.* (2000)] K. Nigam, A.K. McCallum, S. Thrun, and T. Mitchell, 2000. Text Classification from Labeled and Unlabeled Documents Using EM. *Machine Learning*, vol. 39(2), pages 103–134.

[Ng *et al.* (2003)] H.T. Ng, B. Wang, and Y.S. Chan, 2003. Exploiting Parallel Texts for Word Sense Disambiguation: An Empirical Study. *Proceedings of ACL*, pages 455–462.

[Nagao (1994)] K. Nagao, 1994. A preferential constraint satisfaction technique for natural language analysis. *IEICE Transactions on Information and Systems*, Vol. E77–D, No. 2, pages 161–170.

[Ng & Lee (1996)] H.T. Ng and H.B. Lee, 1996. Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplar-Based Approach. *Proceedings of ACL*, pages 40–47.

[Ng (1997)] H. Ng, 1997. Exemplar-Based Word Sense Disambiguation: Some Recent Improvements. *Proceedings of EMNLP*.

[Ngai *et al.* (2004)] G. Ngai, D. Wu, M. Carpuat, C-S. Wang, and C-Y. Wang, 2004. Semantic Role Labeling with Boosting, SVMs, Maximum Entropy, SNOW, and Decision Lists. *Proceedings of Workshop on Senseval-3*, Barcelona.

[Purandare & Pedersen (2004)] A. Purandare and T. Pedersen, 2004. Word sense. discrimination by clustering contexts in vector and. similarity spaces. *Proceedings of the Conference on Computational Natural Language Learning*, pages 41–48

[Pedersen (2000)] T. Pedersen, 2000. A Simple Approach to Building Ensembles of Naive Bayesian Classifiers for Word Sense Disambiguation. *Proceedings of NAACL*, pages 63–69.

[Perrone & Cooper (1993)] M.P. Perrone and L.N. Cooper, 1993. When Networks Disagree: Ensemble Methods for Hybrid Neural Networks. *Neural Networks for Speech and Image Processing*, pages 126–142.

[Pham *et al.* (2005)] T.P Pham, H.T. Ng, W.S. Lee, 2005. Word Sense Disambiguation with Semi-Supervised Learning. *Proceedings of AAAI*, pages 1093–1098.

[Pierce & Cardie (2001)] D. Pierce and C. Cardie, 2001. Limitations of Co-Training for Natural Language Learning from Large Datasets. *Proceedings of EMNLP*, pages 1–9.

[Ratnaparkhi (1996)] A. Ratnaparkhi, 1996. A Maximum Entropy Model for Part-of-Speech Tagging. *Proceedings EMNLP.*

[Ratnaparkhi (1998)] A. Ratnaparkhi, 1998. Maximum Entropy Models for Natural Language Ambiguity Resolution, *Ph.D. thesis*, University of Pennsylvania, Philadelphia, PA.

[Rigau *et al.* (1997)] G. Rigau, E. Agirre, and J. Atserias, 1997. Combining unsupervised lexical knowledge methods for word sense disambiguation. *Proceedings of ACL/EACL97*, Madrid, Spain.

[Shafer (1976)] G. Shafer, 1976. *A Mathematical Theory of Evidence.* Princeton University Press, Princeton.

[Sanderson (1994)] M. Sanderson, 1994. Word Sense Disambiguation and Information Retrieval. SIGIR. *Proceedings of the TOC*, pages 142–151

[Smets (1994)] P. Smets, R. Kennes, 1994. The transferable belief model, *Artificial Intelligence* **66** (1994) 191–234.

[Strapparava *et al.* (2004)] C. Strapparava, A. Gliozzo, and C. Giuliano, 2004. Pattern abstraction and term similarity for word sense disambiguation: IRST at Senseval-3. *Proceedings of Senseval-3 Workshop.*

[Schutze (1998)] H. Schutze, 1998. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–124, 1998

[Sussna (1993)] Sussna, M. 1993. Word sense disamiguation for free-text indexing using a massive semantic network. *Proceedings of the Second International Conference on Information and Knowledge Base Management, CIKM*, pages 67–74.

[Su *et al.* (2004)] W. Su, M. Carpuat, and D. Wu, 2004. Semi–Supervised Training of a Kernel PCA–Based Model for Word Sense Disambiguation. *Proceedings of COLING*, pages 1298–1304.

[Seo *et al.* (2004)] H.C. Seo, H.J. Chung, H.C Rim, S.H. Myaeng, and S.H. Kim, 2004. Unsupervised Word Sense Disambiguation Using WordNet Relatives. *Computer, Speech and Language*, Vol. 18(3), pages 253–273.

[Zhou & Li (2005)] Z-H Zhou , M. Li, 2005. Tri-Training: Exploiting Unlabeled Data Using Three Classifiers. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17(11), pages 1529–1541.

[Tsuruoka & Tsujii (2005)] Y. Tsuruoka and J. Tsujii, Chunk Parsing Revisited. *Proceedings of the 9th International Workshop on Parsing Technologies* (IWPT 2005), pp. 133–140

[Towell & Voorhees (1998)] Towell, G. G., & Voorhees, E. M. (1998). Disambiguating highly ambiguous words. *Computational Linguistics*, 24 (1), 125–145.

[Vapnik (1998)] Vapnik, Vladimir 1998. *Statistical Learning Theory.* New York, USA, John Wiley

[Wilks *et al.*(1993)] Wilks, Y., Fass, D., Guo, C.-M., McDonald, J., Plate, T., and Slator, B. (1993). Providing machine tractable dictionary tools. In Pustejovsky, J. (Ed.), Semantics and the lexicon, pp. 341–401. *Kluwer Academic Publishers.*

[Wang & Matsumoto (2004)] X.J. Wang and Y. Matsumoto, 2004. Trajectory Based Word Sense Disambiguation. *Proceedings of the 20th International Conference on Computational Linguistics*, pages 903–909.

[Wu *et al.* (2004)] D. Wu, W. Su, and M. Carpuat. 2004. A kernel PCA method for superior word sense disambiguation. *Proceedings of ACL.*

[Vapnik (1998)] Vapnik V. 1998. Statistical Learning Theory, Wiley

[Vossen *et al.* (2006)] Vossen P., Rigau G., Alegria I., Agirre E., Farwell D., Fuentes M. 2006 Meaningful results for Information Retrieval in the MEANING project. *Proceedings of Third International WordNet Conference.* Jeju Island (Korea).

[Yager (1988)] R.R. Yager, 1988. On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decision Making, *IEEE Transactions on Systems, Man, and Cybernetics* Vol. 18, pages 183–190.

[Yarowsky (1992)] D. Yarowsky, 1992. Word Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. *Proceedings of COLING*, pages 454–460.

[Yarowsky (1993)] Yarowsky, David. 1993. One sense per collocation. *Proceedings of ARPA Human Language Technology Workshop*, pages 266–271, Princeton, NJ.

[Yarowsky (1994)] Yarowsky, D. (1994). Decision lists for lexical ambiguity resolution: Application to accent restoration in spanish and french. *Proceedings ACL*

[Yarowsky (1995)] D. Yarowsky, 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. *Proceedings of ACL*, pages 189–196.

[Yarowsky *el al.* (2002)] Yarowsky, David and Radu Florian. 2002. Evaluating sense disambiguation across diverse parameter spaces. *Natural Language Engineering* 9(4):293–310.

[Yang & Pedersen (1997)] Y. Yang , J.O. Pedersen, A comparative study on feature selection in text categorization. *Proceedings of ICML*, 14. th. International Conference on Machine Learning, 412-420

[Yager 1988] R. R. Yager, On ordered weighted averaging aggregation operators in multicriteria decision making, *IEEE Transactions on Systems, Man, and Cybernetics* **18** (1988) 183–190.

[Yang & Xu 2002] J. B. Yang, D. L. Xu, On the evidential reasoning algorithm for multiple attribute decision analysis under uncertainty, *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans* 32 (3) (2002) 289–304.

[Zadeh 1983] L. A. Zadeh, A computational approach to fuzzy quantifiers in natural languages, *Computers and Mathematics with Applications* **9** (1983) 149–184.

[Zadeh 1984] L. A. Zadeh, Reviews of Books: A Methematical Theory of Evidence, *The AI Magazine* **5** (1984) 81–83.

[Zhang & Oles (2000)] T. Zhang and F. J. Oles. A probability analysis on the value of unlabeled data for classifi- cation problems. *Proceedings of ICML*, pages 1191–1198, 2000.

[Zheng (2005)] Y.N. Zheng, H.J. Dong, and L.T. Chew, 2005. Word Sense Disambiguation Using Label Propagation Based Semi–supervised Learning Method. *Proceedings of ACL*, pages. 395–402.

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

Publications

Journals

[1] A.C. Le, A. Shimazu, and V.N Huynh, 2006. Word Sense Disambiguation by Combining Classifiers with an Adaptive Selection of Context Representation. *Journal of Natural Language Processing*, Vol 13, No. 1, pp 75–95

[2] A.C. Le, V.N Huynh, A. Shimazu, and Y. Nakamori, 2006. Combining Classifiers for Word Sense Disambiguation Based on Dempster-Shafer Theory and OWA Operators. *Journal of Data and Knowledge Engineering (DKE)* (Accepted).

[3] A.C. Le, A. Shimazu, V.N Huynh, and L.M. Nguyen, 2006. Semi-Supervised Learning Integrated with Classifier Combination for Word Sense Disambiguation. (Revising) *Journal of Computer Speech and Language.*

Referred International Conferences

[4] A.C. Le and A. Shimazu, 2004. High Word Sense Disambiguation using Naive Bayesian classifier with rich features. *The 18th Pacific Asia Conference on Language, Information and Computation (PACLIC-2004)*, pages 105–113

[5] A.C. Le, V.N. Huynh, and A. Shimazu, 2005. Combining Classifiers with Multi-Representation of Context in Word Sense Disambiguation. *The 19th Pacific Asian Conference on Knowledge and Data Mining (PAKDD–2005)*, pages 262–268.

[6] A.C. Le, V.N. Huynh, and A. Shimazu, 2005. An Evidential Reasoning Approach to Weighted Combination of Classifiers for Word Sense Disambiguation. *The International Conference on Machine Learning and Data Mining (MLDM–2005)*, pages 516–525

[7] A.C. Le, V.N. Huynh, H.C. Dam, and A. Shimazu, 2005 Combining Classifiers Based on OWA Operators with an Application to Word Sense Disambiguation. *The Tenth International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC–2005)*,Vol.1 pages 512–521

[8] A.C. Le, A. Shimazu, and M.L. Nguyen, 2006. Investigating Problems of Semi-Supervised Learning for Word Sense Disambiguation, *The 21st International Conference on Computer Processing of Oriental Languages (ICCPOL 2006)*, 12/2006, Singapore.

[9] A.C. Le, V.N. Huynh, A. Shimazu, and H.C. Dam, 2006. Weighted combination of classifiers for word sense disambiguation based on Dempster-Shafer theory. *The International Conference on Research, Innovation and Vision for the Future (RIVF–2006)*, pages 133–138

[10] P.T. Nguyen, Shimazu A., M.L. Nguyen, and A.C. Le, 2006. Rule-Based Transformation for Improving Phrase-Based SMT from English to Vietnamese. The First International Conference on Knowledge, Information and Creativity Support Systems KICSS'06, 1-4 August, Ayutthaya

[11] T.T. Nguyen, , A. Shimazu, A.C. Le, and M.L. Nguyen, 2006. Applying RST Relations to Semantic Search. Ninth International Conference on TEXT, SPEECH and DIALOGUE (TSD-2006)