Technical Report

UCAM-CL-TR-673
ISSN 1476-2986

Number 673

UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Probabilistic word sense disambiguation

Analysis and techniques for combining knowledge sources

Judita Preiss

August 2006

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted July 2005 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Trinity College.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

ISSN 1476-2986

Abstract

This thesis shows that probabilistic word sense disambiguation systems based on established statistical methods are strong competitors to current state-of-the-art word sense disambiguation (WSD) systems.

We begin with a survey of approaches to WSD, and examine their performance in the systems submitted to the SENSEVAL-2 WSD evaluation exercise. We discuss existing resources for WSD, and investigate the amount of training data needed for effective supervised WSD.

We then present the design of a new probabilistic WSD system. The main feature of the design is that it combines multiple probabilistic modules using both Dempster-Shafer theory and Bayes Rule. Additionally, the use of Lidstone's smoothing provides a uniform mechanism for weighting modules based on their accuracy, removing the need for an additional weighting scheme.

Lastly, we evaluate our probabilistic WSD system using traditional evaluation methods, and introduce a novel task-based approach. When evaluated on the gold standard used in the SENSEVAL-2 competition, the performance of our system lies between the first and second ranked WSD system submitted to the English all words task.

Task-based evaluations are becoming more popular in natural language processing, being an absolute measure of a system's performance on a given task. We present a new evaluation method based on subcategorization frame acquisition. Experiments with our probabilistic WSD system give an extremely high correlation between subcategorization frame acquisition performance and WSD performance, thus demonstrating the suitability of SCF acquisition as a WSD evaluation task.

Acknowledgments

I would like to thank my supervisor Ted Briscoe for all the help he has given me during my PhD, especially for proof reading papers at very short notice! I am also indebted to Ann Copestake, my second supervisor, for her feedback on an earlier draft of my dissertation.

The Natural Language and Information Processing research group has provided a stimulating and friendly atmosphere in which to work. I am particularly grateful to Anna Korhonen who explained to me some of the mysteries of subcategorization acquisition, and Aline Villavicencio for encouragement and advice.

There are many individuals to thank for valuable discussions during my PhD: John Carroll for his help and tolerance regarding technical aspects of RASP; Diana McCarthy for helping me clear up numerous WSD terms and encouragement throughout the thesis; and Mark Stevenson for discussing the feasibility of various approaches to WSD.

The financial support of the EPSRC funded RASP project, the Cambridge University Computer Laboratory, and Trinity College are gratefully acknowledged.

My family and friends have provided me with a great deal of encouragement and support, and I would particularly like to thank my parents for always believing in me. Finally, the proof reading skills of my husband Joe greatly improved the dissertation, and without him this thesis wouldn't have been possible.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 The Word Sense Disambiguation Task

Word sense disambiguation (WSD) refers to the task of automatically assigning a sense to a word from a given set of senses. This is motivated by the fact that many words have more than one sense: for example, *pen* can be a writing instrument or an enclosure for animals or children. This sense distinction was introduced into the natural language processing (NLP) literature by Bar-Hillel (1960):

> Little John was looking for his toy box. Finally, he found it. The box was in the pen. John was very happy.

Bar-Hillel concluded on the basis of this example that distinguishing these two senses automatically will always be impossible. He assumed that for disambiguation it would be necessary to represent world-knowledge, such as "toy boxes are smaller than play pens" and "toy boxes are larger than writing pens", making the WSD task AI-complete.

Early approaches focused on carrying out WSD within other application, such as Wilks (1972) who carried out WSD within machine translation. His approach was based on hand-coded selectional preference rules, with the sense assignment being chosen so the largest number of preferences is satisfied. Small's (1980) approach relied on extensive word entries even more heavily – his system assumed that "human knowledge about language is organised primarily as knowledge about words rather than knowledge about rules" (Small and Rieger, 1982).

Such approaches were restricted as to domain and extensibility: although domains could be increased manually, Gale et al. (1992c) pointed out this need to manually encode knowledge as the bottleneck for WSD. Work has since focused on automatically extracting information from existing machine readable corpora. An example, which we focus on within this thesis, are corpus–based methods in which a large amount of text is used to acquire contextual information. This approach has recently yielded promising results.

1.2 Uses of WSD

WSD is not usually seen as an end product, but rather as a component within another system. A system, according to Sparck Jones and Galliers (1996), carries out a task

such as machine translation or information extraction. A system can be evaluated in its own right by non-linguists, for example by judging the comprehensibility of a translated text. These systems use components, examples being part of speech taggers, or word sense disambiguation. The output of components is rarely interesting to a user directly, but a component may create a user-detectable change in performance of a system which incorporates it.

A number of tasks have been shown to benefit from employing WSD. We summarized three of these below, for further systems, such as information retrieval or text-to-speech synthesis, see e.g., Stevenson (2003). We focus on machine translation (e.g., Brown et al. (1991)), accent restoration (e.g., Yarowsky (1996)) and verb subcategorization acquisition (Korhonen and Preiss, 2003), and illustrate these applications with examples:

Machine Translation: In this task, a system is required to translate from one language into another. In a Czech to English machine translation system, the Czech word *pánev* can be translated as either *frying pan* or *pelvis*. Thus information about sense is important when translating a sentence such as the following:

> *Udělám omeletu na pánvi.*
> (I will) make an omelet with the frying pan.

For example, Brown et al. (1990) use corpus alignment to produce language and translation models, which combine to generate a probability for an English sentence translation given the French sentence original. They obtain an improvement over a naive method based purely on a priory word probabilities.

Accent Restoration: Accents (which affect pronounciation and meaning) may be removed by computer processing of texts, so we rely on an accent restoration system to put these back. Compare the following Czech sentences:

> *Kdy mámě dal síť?*
> When did (he) give mother a net?

and

> *Kdy máme dál šít?*
> When should (we) carry on sowing?

The two sentences have an identical form when their accents are stripped, but a vastly different meaning. Of course, most of the time the distinction will only be between accents on one word, and local syntactic patterns and collocational information can be used to restore them (Yarowsky, 1994).

Verb Subcategorization Acquisition: Different verbs require various arguments (subcategorization frames, SCFs) in order to form grammatically correct sentences (Levin, 1993). In fact, different senses of a verb may require different SCFs. For example, the 'accepting as true' sense of believe in the sentence

> *I believed his report*

takes an NP complement, whereas the 'religious' sense of *believe* in the following sentence is intransitive.[1]

When you hear his sermons, you will be able to believe, too

WSD is the task of choosing a sense for a word from a given set of senses, however, the set of senses is not always the same (e.g, Wilks and Stevenson (1996), or Resnik and Yarowsky (1997)). E.g., in the accent restoration example above, *mame* has two senses: *mámě* or *máme*. In fact, Pustejovsky (1995) argues that enumerative dictionaries are insufficient, as words can take on infinitely many meanings by being used in novel contexts (for a discussion of the notion of sense, see Section 3.2.1). As Palmer et al. (2002) hypothesises that a more fine-grained lexicon leads to a more difficult WSD task, it is important to specify the tasks which we investigate in this work. We distinguish two types of WSD tasks:

Inventory–based task in which we employ a given sense inventory taken from a lexicon created independently of WSD (e.g., a machine readable dictionary such as WordNet (Miller et al., 1990)). A WSD system in this framework is scored directly against a gold standard markup of the text on which it is run, and its output is not used further. However, even machine readable dictionaries vary in fine-grainedness due to the lexicographers' instructions and intuition (Kilgarriff, 1997). In this work, we use WordNet 1.7 for inventory-based evaluations. Although it has been criticized for being very fine-grained, it was designed for research purposes, is freely available and so can easily form the basis for a shared sense inventory (Kilgarriff, 2002).

Application–based task where the application dictates the fine-grainedness of the sense distinctions necessary (for example, the sense distinctions necessary for the subcategorization acquisition task are very coarse (Korhonen, 2002)). In this task, the accuracy of the WSD system is measured by the change in performance of the end product. In Section 7.5 we investigate how sensitive subcategorization acquisition is to the accuracy of the WSD system.

1.3 Knowledge–base Alternatives

This dissertation explores alternatives to the knowledge based approaches such as corpus–based methods. Systems exploiting this sort of information are of two types: supervised or unsupervised. A supervised approach usually involves acquiring some patterns from a large body of annotated text (e.g., Mihalcea (2002), or Yarowsky and Florian (2002)), often using machine learning techniques (e.g., Pedersen (2002), or Hoste et al. (2002)). In this case, the text used for acquiring patterns (the *training* data) is manually sense tagged. However, it is difficult to obtain large sense–tagged corpora: often trained lexicographers are required and the process is lengthy and expensive. Unsupervised approaches do not use sense–tagged data (and therefore do not have a separate training phase). For example, choosing senses which maximize the overlap of words in their definitions is an unsupervised approach which exploits a machine readable dictionary (Lesk, 1986) (see Section 2.4 for an example).

[1] These example sentences are taken from WordNet 1.7 (Miller et al., 1990).

Usually, supervised systems outperform unsupervised systems (see Section 2.5.2), and therefore we focus mainly on supervised algorithms in this work. In particular, we are interested in approaches which acquire patterns from a body of sense–tagged text. This approach is based on the idea that similar contexts indicate particular senses of words; seeing the word *money* near the word *bank* strongly suggests the financial institution sense of *bank*, indicating that co-occurrence patterns may be useful in WSD.

It has been observed that sense assignments are usually more accurate when multiple information sources are combined (Stevenson and Wilks, 2001). For example, it is useful to know both the part of speech of the word *bank* (this word also has a verb sense) and the co-occurrence information with the word *money*. However, nobody has yet found a provably optimal method for combining information sources. For example, Stevenson and Wilks (1999) initially used majority voting to combine their information sources, whereas Yarowsky (2000) used hierarchical decision lists. We present a new application of the Dempster-Shafer theory (Shafer, 1976) – a principled method for combining information sources based on each information source producing a probability distribution on senses. We now motivate the need for generating probability distributions and therefore also the need to combine these.

Many systems generate a single (forced) sense assignment for each word, and it is not clear that this is always optimal: even in a human sense annotation there are words for which the annotators cannot choose just one sense. For example, the word *peculiar* in the following sentence (from the English all words task in SENSEVAL-2):

The art of change-ringing is peculiar to the English ...

can be assigned either of the two senses:

Sense	Definition	Example sentence
characteristic	characteristic of one only; distinctive or special	"the peculiar character of the Government of the U.S."- R.B.Taney
specific	unique or specific to a person or thing or category	rights peculiar to the rich

In this case, generating a probability distribution over the available senses may be more useful. Such a distribution can also be used in other applications of WSD: for example, the back-off estimates in subcategorization acquisition can be guided by the senses present in the corpus (Preiss and Korhonen (2002), and Section 7.4.2). A forced choice, single sense assignment can be easily obtained from a probability distribution, so there is no loss of information in producing a probability distribution.

Each of our information sources therefore produces a probability distribution on senses, which is smoothed according to our confidence in the given information source. An overall probability distribution on senses is given by a combination of all the information sources' probability distributions. We investigate the relative performance of the system when theoretically motivated methods for module combination (such as Dempster-Shafer theory or Bayes Rule) are used compared to the performance based on using weighted linear interpolation.

1.4 Contributions of this Thesis

In this thesis we present the following innovative research:

- We design a modular probabilistic WSD system, for which we adapt a number of existing WSD approaches to function individually, each producing a probability distribution on senses.
- We apply WSD to the task of SCF acquisition and find a very high correlation between SCF acquisition performance and WSD performance, resulting in a new task-based method for evaluating WSD systems.
- We apply the Dempster-Shafer theory in the WSD domain to combine information sources and to produce an overall probability distribution on senses. The performance of the WSD system when Dempster-Shafer theory is used to combine information sources is compared to the performance when Bayes Rule, and weighted interpolation are employed in the combination.
- We use Lidstone's smoothing to provide a uniform mechanism for weighting modules based on their accuracy.

1.5 Thesis Overview

Although we do not attempt to provide a complete overview of WSD algorithms, Chapter 2 surveys past and present approaches to WSD, and examines their performance in the systems submitted to the English all words task in SENSEVAL-2.

The discussion of the two evaluation methods for WSD, task-based or inventory–based evaluation, forms a large part of this work and is presented in Chapter 3. In this chapter, we also explain the difficulty of obtaining sense annotated data, and investigate how the size of the training corpus affects the performance of a WSD system.

The following chapter, Chapter 4, is a detailed study of 25 systems which motivates the modular approach taken in this thesis. We present the design of a new probabilistic WSD system in Chapters 5 and 6. The main feature of the system is that it is composed of multiple probabilistic components: such modularity is made possible by the application of Dempster-Shafer theory, Bayes Rule and Lidstone's smoothing method.

We describe our novel task-based method for evaluating WSD based on subcategorization acquisition in Chapter 7, in which we also show that the subcategorization frame acquisition task is suitable for evaluating WSD systems.

In Chapter 8 we evaluate our system on the SENSEVAL-2 English all words task: in its raw form it would appear second in the list of results. The various combination methods are compared, and the results on the SENSEVAL-2 English lexical sample task are also presented.

We draw our conclusions in Chapter 9, and summarize the contributions of this thesis. We also suggest some ideas for future work.

Chapter 2

Approaches to WSD

2.1 Introduction

This chapter contains descriptions of a number of approaches to WSD, many of which form the basis of our work. It is not intended as a historical overview or a complete list of WSD systems, rather such surveys can be found elsewhere. We discuss the 1950s beginnings of WSD, and provide short overviews of three main types of knowledge-based approaches to WSD, illustrating each type with an existing probabilistic system. We motivate the need for baselines and present the most commonly used baselines. We introduce the SENSEVAL evaluation exercise, and summarize current state-of-the-art WSD systems.

2.2 Beginnings of WSD

This section introduces the beginnings of WSD – it is not intended to be a complete list of WSD systems, for such surveys see for example Ide and Véronis (1998), Jurafsky and Martin (2000), or Manning and Schütze (1999), or, for more recent approaches, Preiss and Yarowsky (2002), Edmonds and Kilgarriff (2002), or Mihalcea and Chklowski (2004).

WSD was initially discussed in the context of machine translation (MT) – Weaver (1955) pointed out the need for WSD within MT as different senses may have different representations in the target language. He observed that it may not be possible to disambiguate a word independent of context, but one can establish the meaning of the word if N words either side are known. This approach forms the basis of many WSD algorithms today.

Many current approaches were initially sketched out in the 1950s and 1960s, however as they were not implemented due to a lack of resources, they were often forgotten. Weaver also pointed out that the number of senses a word has may be reduced within some domains – this idea was extended by Madhu and Lytle (1965), who considered the sense frequencies for different domains. They obtained probabilities of each sense given the context using Bayes' Rule.

A number of approaches from the AI (understanding) point of view were investigated in the 1960s. Masterman (1961) used semantic networks to represent sentences in an interlingua; she created 100 primitive concepts (such as THING) which in turn yielded a concept dictionary where concepts were ordered in a hierarchy. A sentence representation was chosen so the nodes in the network were as close as possible. The work of

Quillian (1962) extended this work, by including extra links between words and concepts. WSD was performed by finding the concept which linked the input words most directly, which is a method frequently employed in contemporary dictionary based approaches. Masterman's concepts were used by Wilks (1968), whose preference semantics employed selectional restrictions, however these could be relaxed when the preferred restriction did not appear.

In 1965, Bar-Hillel presented examples which, according to him, computers were never going to accurately disambiguate. These remarks indicated that WSD was an AI-complete problem, and formed part of the ALPAC report (1966), which lead to an end of much of the research on MT. There was therefore a decrease of interest in WSD; research continued mainly on natural language understanding systems based on semantic network knowledge representation, which were applied to WSD (e.g., Quillian (1969), or Hayes (1976)), with more projects starting up in the 1980s.

Our work focuses on knowledge-based approaches which we describe in more detail in the following sections. Specifically, we augment recent knowledge-based approaches to fit into our probabilistic model, and investigate combination methods.

2.3 Knowledge-based Approaches

Although there are many differences between various WSD systems (for example, varying the tagger employed by a WSD algorithm may have an effect on performance), existing knowledge-based WSD systems can be grouped into three main types (Jurafsky and Martin, 2000). We will describe them and provide examples for each approach, restricting ourselves to existing statistical approaches since these are the closest to our work. For a more detailed description of any of the approaches, see e.g., Charniak (1993).

2.3.1 Semantic Approaches

These approaches encode information about the restrictions imposed by certain words, they are not necessarily directly supervised but often rely on the use of other tools such as parsers or hand coded information. For an example, consider the following Wall Street Journal sentences illustrating the preference of the word *dish* (Jurafsky and Martin, 2000):

1. "In our house, everybody has a career and none of them includes washing *dishes*," he says.

2. In her tiny kitchen at home, Ms. Chen works efficiently, stir-frying several simple *dishes*, including braised pig's ears and chicken livers with green peppers.

The words *wash* and *stir-fry* impose restrictions on the word *dishes* allowing us to distinguish the two different senses (the physical objects we eat from, and the actual meals, respectively).

Katz and Fodor (1964) associated hand coded semantic markers with every word in their lexicon, which detailed the conditions under which two words could combine. Katz's approach was automated for example by McCarthy (2003) who used a wide-coverage parser producing head–dependent relations (Briscoe and Carroll, 2002) to populate the WordNet hierarchy. They then acquire a tree cut model (Li and Abe, 1998) conditioned

on a verb class, and either the subject or direct-object relation, or an adjective-class and the adjective-noun relation. Disambiguation is performed by finding the sense with the maximum probability given the context.

This method suffers from an obvious sparse data problem, since it involves knowing grammatical information such as the possible direct objects of each verb. It therefore usually attains low recall and is not easy to scale up (see the performance breakdown of the Sussex-sel system in Table 2.6).

Example

To reduce the sparseness problem, Resnik (1992) used the WordNet hierarchy to define groups of words: he groups words according to the ability to substitute them in context. His selectional restrictions provide information about groups of words (rather than individual words), and he selects the sense contained in the group with the greatest mutual information content, thus increasing recall.

2.3.2 Corpus–based Approaches

These approaches often make use of feature vectors representing contextual information about the target word, such as the word to the left of the target word or whether a particular word occurs within a k–sized window of the target word. Such feature vectors can then be used by a machine learning algorithm.

As corpus–based approaches can be further classified depending on the amount of sense annotated text (training data) they employ, we will use this opportunity to clarify the definitions of supervised and unsupervised systems. A supervised system makes use of a disambiguated corpus for training. The training corpus contains occurrences of words with their associated senses. New words are then classified according to their 'similarity' to the words in the training corpus. Unsupervised systems do not use such a corpus. Corpus–based approaches can be classified in the following way:

1. Supervised systems start off with sense tags associated with the feature vectors, which are drawn from an annotated corpus. These are then input to a machine learning algorithm, such as naive Bayes (e.g., Pedersen (2000)), decision tree learning (e.g., Yarowsky (2000)) or nearest neighbour methods (e.g., Veenstra et al. (2000)), which generalize from the training instances to classify new examples. This approach usually requires a large amount of training data.

2. Bootstrapping only requires a small number of, usually manually created, training examples and the system itself extracts a larger training set of similar examples from the untagged corpus (e.g., Mihalcea and Moldovan (2001)). The larger training set is often biased towards the initial examples, not producing a balanced training corpus.

3. Unsupervised systems do not need to start off with any sense tagged data and they use clustering methods to group together 'same' senses. However, it is not always clear what the resulting clusters correspond to, as they rarely have an obvious connection to senses identified by lexicographers listed in dictionaries and therefore this is more of a discrimination task.

Examples

Ng et al. (2003) explore the possibility of using word-aligned parallel corpora to automatically generate training data for their WSD system. They restrict their sense inventory to the distinctions occurring between English and Chinese for the words they are interested in. They use the aligned corpus to create feature vectors for each occurrence-sense pair which describe parts of speech, surrounding words etc. The naive Bayes algorithm is used to find the sense assignment.

The following two heuristics are widely used within corpus–based approaches due to their ability to boost recall (Yarowsky, 1995).

1. The one-sense-per-collocation approach is based on the observation that within collocations (by collocation we mean a frequently used co-occurrence of words), words usually only have one sense. For example, the word *tea* in *iced tea* will usually mean the beverage rather than the mid-afternoon meal. We only therefore need to disambiguate the collocations once and can use this sense annotation whenever we observe the collocation.

2. The one-sense-per-discourse approach works on a similar principle. Experimenting on encyclopedia articles, Yarowsky found that words usually have only one sense throughout a document and therefore only need to be resolved once per document. Krovetz (1998) suggests that this hypothesis does not necessarily hold once you move to fine-grained senses, however the recent work by Koeling et al. (2005) demonstrates that one sense usually dominates within a restricted domain.

2.3.3 Dictionary–based Approaches

These approaches only use information derivable from some machine readable dictionary. The first example of a dictionary–based approach was Lesk's (1986) definition overlap; the main idea of this approach is that related words will also have words in common in their definitions. This approach therefore chooses those senses that maximize overlap between the definitions of words within a 10 word window of context (for an example, see Table 2.4). A drawback of this method is that dictionary definitions are often quite short and the presence or absence of one word can make a large difference.

Example

Véronis and Ide (1990) extend Lesk's dictionary overlap approach, by building very large neural networks from definitions in machine readable dictionaries. The nodes in these neural networks represent words (or concepts) connected by links which can be activated. When activated with words from a sentence, the network iteratively activates nodes until the process stabilizes. The most activated senses are chosen after a number of cycles. The authors overcome the usual need to hand code neural networks by using definition words to produce the microfeatures representing each word. Their system outperforms the underlying Lesk algorithm.

2.4 Baselines

It was noticed by Gale et al. (1992b) that different systems may not be immediately comparable. Firstly, systems may differ in the number of words they were designed to attempt (e.g., Bruce and Wiebe (1994) evaluated their system on the single word *interest*, whereas the system of Stevenson (1999) was evaluated on all labeled words in the SEMCOR corpus). Secondly, systems used different evaluation corpora which can lead to large performance differences. Thirdly, systems were based on different sense inventories.[1] WSD precision of 90% is usually very good (the highest performing system on the SENSEVAL-2 English all words task has an F-measure of 68.9%), but only if there are many ambiguous words in the corpus with a number of highly frequent senses (otherwise these would be accurately resolved using the most frequent sense heuristic).

Gale et al. therefore suggested the need to include a lower and upper bound for a text together with performance results. An upper bound, the measure of agreement among human annotators, indicates the highest possible accuracy a system could achieve – it would be unreasonable to expect a system to accurately tag senses for which humans were not confident. A lower bound represents the performance of an extremely basic system; we now summarize the three most commonly used lower bound (baseline) systems.

The **most frequent sense** (Gale et al., 1992b) baseline assigns to each word its most frequent sense. This most frequent sense is usually drawn from a different sense annotated corpus (such as SEMCOR) or domain analysis can be performed to find the most frequent sense of each word in the current text (Koeling et al., 2005). Note that knowing the most frequent sense of a word in the current corpus would involve sense tagging the whole corpus (and thus having an accurate WSD system). The most frequent sense approach works surprisingly well, due to the Zipfian distribution on senses (e.g., Sanderson and Rijsbergen (1999)). To illustrate this baseline on a running example, consider the word *affair*, which has the following noun senses in WordNet:

Rank	Id	Sense
1	affair%1:04:00::	a vaguely specified concern
2	affair%1:11:00::	a vaguely specified social occasion
3	affair%1:26:00::	a usually secretive or illicit sexual relationship

The most frequent sense baseline will assign the top ranked sense (a vaguely specified concern) to the word in both of the following sentences, even though the second sentence corresponds to the second ranked meaning of the word *affair*:[2]

1. It is none of your affair.
2. The party was quite an affair.

Lesk (1986) overlap considers a 10 word window and selects those senses which correspond to the largest overlap between their dictionary definitions. Useful especially

[1] A more fine-grained sense inventory will mean that the WSD task is probably more difficult and therefore will lead to lower performance.

[2] Example sentences taken from WordNet.

in learners' dictionaries, where only a small set of words is used in dictionary definitions (e.g., 2000 words in CIDE+ (Procter, 1995)), this system relies on related concepts sharing topical words in their definitions. We use the second sentence above to illustrate this method. In this sentence we need to resolve the nouns *party* and *affair*. The definitions for *affair* are given above and here we can find the definitions for the noun *party*:

Rank	Id	Sense
1	party%1:14:01::	an organization to gain political power
2	party%1:11:00::	an occasion on which people can assemble for social interaction and entertainment
3	party%1:14:02::	a band of people associated temporarily in some activity
4	party%1:14:00::	a group of people gathered together for pleasure
5	party%1:18:00::	a person involved in legal proceedings

We can see that the largest overlap of definitions will come from the second senses of both words (the definitions share the word *social*) and therefore these senses will be chosen.

The **Random** baseline uses a uniform distribution over senses to choose one sense for each word.

2.5 The SENSEVAL Exercise

The suggestion of Gale et al. (1992b) to cite performance results along with upper and lower bound measures for the WSD task still does not provide enough information for a meaningful comparison of systems. For instance, one system may select a completely unrelated sense (a homonym) whereas another may return a related, but wrong, sense. The second system may turn out not to be making such a serious error if the output were used for machine translation into a language where the two senses map to the same target word.[3]

Resnik and Yarowsky (1997) extended the work of Gale et al. (1992b) to introduce the notion of a WSD evaluation task. Aside from demonstrating the need for such a task, they also provided a format for evaluation and brought up problems which needed to be addressed. They argued that an evaluation exercise would ensure that results are comparable across systems, and future detailed comparisons (such as closeness of answers) would be possible. Their work resulted in the SENSEVAL evaluation exercise (Kilgarriff, 1998). There have been two SENSEVAL evaluation exercises up to 2004, and descriptions of the tasks available at these can be found in Table 2.1. Since the sense inventory in SENSEVAL-1 was Hector (Atkins, 1992–93), which is quite a coarse-grained dictionary and not freely available, we do not concern ourselves with this exercise here.

[3]If the WSD system was originally designed only for MT between two languages, it would only distinguish those senses which have a different representation in the target language.

	SENSEVAL-1	SENSEVAL-2
Date held	Summer 1998	Summer 2001
Lexical sample	English, French, Italian	Basque, English, Italian, Japanese, Korean,Spanish, Swedish
All words		Czech, Dutch, English, Estonian
Translation		Japanese

Table 2.1: Description of the tasks available in SENSEVAL

There were three types of tasks in various languages: lexical sample, all words and translation tasks. For the organizers the lexical sample task involves preselecting a set of ambiguous words and tagging a number of their occurrences (in SENSEVAL-2 the number of occurrences tagged for each word was $75 + 15n$, where n is the number of senses the word has, and these instances were randomly chosen). A part of the tagged corpus is given to participants to use as training data for supervised systems, the remainder is tested on. For this task, participating systems frequently train "word experts": the system may be functioning differently depending on which word is being resolved, and so a slightly different version of the system may exist for each word. For a description of the SENSEVAL-2 English lexical sample task see Section 2.5.1.

An all words task consists of organizers' manually tagging all content words (nouns, verbs, adjectives, and adverbs) in at least 5000 words of continuous text. The systems participating in this task are less likely to train "word experts", as training data may not be available for all words. For a description of the SENSEVAL-2 English all words task see Section 2.5.2. The translation task will be discussed in Section 7.2.2, along with our task–based method for evaluating WSD.

The main difficulty of gold standard disambiguation (lexical sample or all words tasks) is deciding on a method to score systems. SENSEVAL implements the metrics of precision and recall:

$$\begin{aligned} \text{precision} &= \%\text{ of right answers in the set of answered instances} \\ \text{recall} &= \%\text{ of right answers on all instances in the test set} \end{aligned}$$

which leads to a combined value of F-measure:

$$\text{F-measure} = \frac{2*\text{precision}*\text{recall}}{\text{precision}+\text{recall}}$$

However, it is not clear what a 'right answer' is – intuitively, we do not want to penalize systems as much for selecting a related sense as we do for selecting a sense which is entirely wrong. For example, Table 2.2 contains three senses of the word *night*. Intuitively, the first two senses appear closer to each other than either of them compared to the last sense. Resnik and Yarowsky (1999) suggest that a measure based on cross-entropy or perplexity would allow for the case where a number of very fine-grained senses are essentially correct. However, the measures discussed by Resnik and Yarowsky are based on a WSD system producing a probability distribution on senses, which is not yet very frequent for WSD systems. This is the main reason for our choice of a probabilistic WSD system, as it can assign a high probability to all close senses rather than choosing one through a forced

choice method.

Id	Definition
1:28:02::	the dark part of the diurnal cycle considered a time unit
1:28:00::	the time after sunset and before sunrise while it is dark outside
1:26:00::	darkness

Table 2.2: Senses of the word *night*

Gold standard	System answers	Score
art_1	art_1	1
bell_ringing_1	–	–
change_1	$\text{change}_1(\mathbf{\frac{1}{2}})$ $\text{change}_2(\mathbf{\frac{1}{2}})$	$\frac{1}{2}$

Table 2.3: Toy corpus: precision $= \frac{1+\frac{1}{2}}{2} = \frac{3}{4}$ and recall $= \frac{1+0+\frac{1}{2}}{3} = \frac{1}{2}$

The scoring method for SENSEVAL-2 did allow for scoring systems producing more than one sense. For an example, see Table 2.3, where we illustrated partial credit obtained from returning a number of senses with varying probabilities (the numerator is the sum of the probabilities assigned to the correct senses).

In this work, we use the English lexical sample task and the English all words task of SENSEVAL-2 to evaluate our WSD system. The following sections provide short descriptions of each of the tasks as well as performance figures and an outline for the participating systems.

2.5.1 English Lexical Sample Task

The English lexical sample task in SENSEVAL-2 contained 29 nouns, 15 adjectives and 29 verbs. In total, 5266 instances were labeled for nouns, 2301 for adjectives and 5373 for verbs. These were divided in a 2:1 ratio into training data, which was supplied to participants in advance, and test data. However, not all 25 participating systems were supervised, see Table 2.4 for performance information. The second column of this table contains a tick if the system is supervised.

We now describe two of the highest performing supervised systems and a short summary of all systems participating in the English Lexical Sample task can be found in Table 2.5. Our system is based on and extends the ideas of numerous past approaches among them the following two.

Under the original scoring system,[4] the highest performing system was **SMUls**, a system which combines a pattern learning module with active feature selection. The sense tagged corpora SEMCOR, WordNet definitions and GENCOR (this is a sense tagged corpus created automatically by the authors of the SMUls system (Mihalcea and Moldovan,

[4]Systems were evaluated twice: once prior to the SENSEVAL workshop (and therefore by the original deadline), but participants were allowed to subsequently resubmit answers if the authors had found performance altering bugs in their programs which lead to a second evaluation. It is the resubmitted results which are presented in Table 2.5.

System	S	Precision	Recall	F-measure
JHU (revised)	√	64.2%	64.2%	64.2%
SMUls	√	63.8%	63.8%	63.8%
KUNLP	√	62.9%	62.9%	62.9%
Stanford-CS224N	√	61.7%	61.7%	61.7%
Sinequa-LIA	√	61.3%	61.3%	61.3%
Duluth3	√	57.1%	57.1%	57.1%
TALP	√	59.4%	59.4%	59.4%
BCU-ehu-dlist-all	√	57.3%	56.4%	56.8%
UMD-SST	√	56.8%	56.8%	56.8%
Duluth5	√	55.4%	55.4%	55.4%
DuluthC	√	55.0%	55.0%	55.0%
Duluth4	√	54.2%	54.2%	54.2%
Duluth2	√	53.9%	53.9%	53.9%
Duluth1	√	53.4%	53.4%	53.4%
DuluthA	√	52.3%	52.3%	52.3%
DuluthB	√	50.8%	50.8%	50.8%
UNED-LS-T	√	49.8%	49.8%	49.8%
WASP	×	58.1%	31.9%	41.2%
Alicante	√	42.1%	41.1%	41.1%
UNED-LS-U	×	40.2%	40.1%	40.1%
Most frequent sense	×	39.4%	38.7%	39.1%
BCU-ehu-dlist-best	√	82.9%	23.3%	36.4%
ITC-irst	×	66.5%	24.9%	36.2%
DIMAP	×	29.3%	29.3%	29.3%
IIT2	×	24.7%	24.4%	24.5%
IIT1	×	24.3%	23.9%	24.1%

Table 2.4: Fine-grained performance on the English lexical sample task

UMD Supervised	Uses features based on wide context as well as local information together with position. Classification is performed using the support vector machines learning framework.
Sinequa-LIA Supervised	Three different window sizes are employed for context based semantic classification trees. The sense assignment is performed using a similarity distance.
TALP Supervised	LazyBoosting used to combine local, topical and domain information. Reduction of senses by hierarchical decomposition of the multiclass problem.
UNED-LS -U Unsupervised -T Supervised	Computes a matrix of mutual information for a fixed vocabulary and applies it to weight co-occurrence counting between sense and context characteristic vectors.
IIT1, IIT2 Unsupervised	Maximize the amount of overlap between target word's WordNet example sentences and its context.
Stanford-CS224N Supervised	23 supervised classifiers (Naive Bayes, vector space, memory-based and other classifier types) combined using one of majority voting, weighted voting or a maximum entropy model.
DIMAP Unsupervised	Uses WordNet to exploit contextual clues: focuses on collocation patterns, contextual overlap with definitions and examples, and topical area matches.
ITC-irst Supervised	Uses WordNet semantic domains to initially determine a word's domain. The similarity between a domain vector and training data domain vectors for the same lemma is used to select a sense.
BCU-dlist-ehu Supervised	Both methods are based on Yarowsky's decision lists. *BCU-dlist-ehu-all* is trained using local and global features, whereas *BCU-dlist-ehu-best* only uses the best features for each word.
SMUls Supervised	Combine instance based learning with an optimized feature selection scheme, creating meta word experts.
Alicante Supervised	Combines a classifier which uses the hyponymy/hypernymy relations in WordNet with maximum entropy probability models.
Duluth Supervised	A number of learnt decision trees and Naive Bayesian classifiers, where the features are the context of the ambiguous word.
KUNLP Supervised	Classification information model which uses local, topical and bigram contextual features.
WASP Unsupervised	Uses Yarowsky's decision lists with manually labeled salient collocates.
JHU Supervised	Voting based classifier combination using many different systems (decision lists, cosine-based vector model and Bayesian classifiers).

Table 2.5: Lexical sample task system descriptions

1999)) were used to acquire patterns for each sense tagged word. A pattern consists of information about the local context of the sense tagged word, such as the words' base forms, their part of speech, WordNet hypernyms if these are known and frequency information. For example, a pattern for the phrase *clear water*, which could be used to disambiguate either *clear* or *water*, may look like this:

clear /	*JJ* /	*4*	*water* /	*NN* /	*1*
base form	PoS	frequency	base form	PoS	frequency

All matching patterns are extracted for a target word, and it is the strongest pattern which will indicate the sense to be chosen.

However, within the lexical sample task, there is another phase of this algorithm. If no pattern applies to resolving the sense of the target word, the algorithm switches to instance based learning with active feature selection. A number of features (such as the word itself, its part of speech, surrounding words and their part of speech, collocations, keywords in context) are employed within an instance based algorithm with information gain feature weighting. Starting with an unordered bag containing all features, the training data is used to find the feature which leads to the best accuracy. This feature is removed from the bag of features and is considered the most informative. The algorithm is repeated to find the second most informative feature, and so on. To assign a sense, the features are applied in the order generated until a decision is made.

The second system we describe is **JHU**, which resulted in the highest performance after resubmission. This system is also based on using multiple features, such as raw words, lemmas, parts of speech, and various positioned relationships to the target word. These features are used to create vectors for each instance in the training data, and are grouped into classes according to their classification. Each class' centroid is computed and its similarity is compared to each vector from the test data. The most similar sense is assigned.

This vector-based model gives rise to three distinct classifiers, based on the similarity measure used. A fourth classifier is based on interpolated decision lists (Yarowsky, 2000). In a decision list, features are ordered according to their smoothed log of likelihood ratio for each sense and are applied in the resulting order to provide a sense assignment. To assign a sense, the result of the decision list is used if the system has a high confidence in this, otherwise majority voting is used to select a sense from the classifiers.

2.5.2 English All Words Task

For the English all words task in SENSEVAL-2, nouns, verbs, adjectives and adverbs in three given texts (taken from the Wall Street Journal) were manually annotated using the WordNet 1.7 pre-release to create a gold standard. This resulted in 1082 distinct words in the texts, corresponding to 2473 instances to be labeled[5] with senses by participating systems. Twenty one systems sense-annotated the given texts, and submitted their answers. As training data was not available for this task, supervised systems had to make use of other annotated corpora (see Section 3.3). Performance information can be seen in Table 2.6.

[5]This includes 86 instances labeled with the "U" tag. This label was assigned when the correct sense of a word did not appear in WordNet 1.7 pre-release.

System	S	Precision	Recall	F-measure
Correct sense	–	100.0%	100.0%	100.0%
SMUaw	√	68.9%	68.9%	68.9%
Most frequent sense	–	66.9%	64.6%	66.7%
CNTS-Antwerp	√	63.5%	63.5%	63.5%
Sinequa-LIA-HMM	√	61.8%	61.8%	61.8%
UNED-AW-T	×	57.4%	56.8%	57.1%
UNED-AW-U	×	55.5%	54.9%	55.2%
IRST	√	74.7%	35.7%	48.3%
UCLA-gchao	√	50.8%	44.4%	47.4%
UCLA-gchao2	√	48.3%	45.3%	46.8%
UCLA-gchao3	√	48.1%	45.1%	46.6%
DIMAP	×	45.1%	45.1%	45.1%
BCU-ehu-dlist-all	√	57.2%	29.1%	38.6%
USM2	×	36.0%	36.0%	36.0%
Random sense	–	35.5%	34.2%	34.8%
USM3	×	33.6%	33.6%	33.6%
USM1	×	34.2%	31.6%	32.8%
Sheffield	×	44.5%	20.0%	27.6%
Sussex-sel-ospd	×	56.6%	16.9%	26.0%
Sussex-sel-ospd-ana	×	54.5%	16.9%	25.8%
Sussex-sel	×	59.8%	14.0%	22.7%
IIT2	×	32.8%	3.8%	6.8%
IIT3	×	29.4%	3.4%	6.1%
IIT1	×	28.7%	3.3%	5.9%

Table 2.6: Fine-grained performance on the English all words task

BCU-ehu-dlist-all Supervised: SEMCOR 1.6	Based on Yarowsky's decision lists, learns bigrams and trigrams for lemmas, word forms and PoS from training data. Also acquires context words from a ±4 word window.
CNTS-Antwerp Supervised: SEMCOR 1.6	A number of machine-learning word experts are trained, and the best one (based on training data) is individually selected for each word-PoS combination.
DIMAP Unsupervised	Uses WordNet to exploit contextual clues: focuses on collocation patterns, contextual overlap with definitions and examples, and topical area matches.
IIT Unsupervised	*IIT1* uses WordNet examples and synsets to choose the sense which has the largest overlap with the instance's context. *IIT2* reduces contributions to score from distant context words. *IIT3* optimizes using the chosen senses for words preceding the current instance.
IRST Unsupervised	Uses WordNet semantic domains to initially determine a word's domain. The similarity between a domain vector and training data domain vectors for the same lemma is used to select a sense.
SMUaw Supervised	Training corpus: SEMCOR 1.6, examples from WordNet 1.7, own heuristically-created sense-tagged corpus. System uses training data to learn a number of patterns from local context.
Sheffield Unsupervised	Restricted to nouns, where it chooses senses by minimizing WordNet distance using simulated annealing. Extra information is provided by an anaphora resolution preprocessing step.
Sinequa-LIA-HMM Supervised: SEMCOR 1.6	Uses WordNet semantic classes as well as short range context to acquire semantic classification trees from training data. Combined with a long-range similarity measure.
Sussex-sel Unsupervised	Identifies subject–verb and verb–direct object relationships and disambiguate these words using class-based selectional preferences. *Sussex-sel-ospd* and *Sussex-sel-ospd-ana* implement the one sense per discourse heuristic. *Sussex-sel-ospd-ana* also uses anaphora resolution to increase coverage.
UCLA-gchao Supervised: SEMCOR 1.6	Creates a probabilistic network for each sentence, modeling dependencies between words. The parameters are obtained automatically and the systems differ in the way these are smoothed. The senses for the sentence are chosen by performing a query over the network.
UNED Unsupervised	Uses mutual information and co-occurrence information along with some frequency heuristics to select a sense.
USM Unsupervised	Partially disambiguated definitions in WordNet and used a semantic distance matrix between senses to make a choice. The three systems are different in the number of words they use from the definitions.

Table 2.7: All words task system descriptions

We describe the top two supervised systems in more detail, Table 2.7 presents short system descriptions for all systems including the training corpus if one was used. The training data was often inadequate (number of instances was too low) and so the performance gap between supervised and unsupervised systems is smaller than in the lexical sample task.

The most successful system was **SMUaw** and the English lexical sample task version of this system was described in Section 2.5.1. The all words system is identical, except the active feature selection is not used as the number of training instances does not reach the minimum needed for the instance based learning.

The second highest performing system was **CNTS-Antwerp**, which trained three classifiers using machine learning. Memory based learning is employed to classify target words according to the similarity of their local context to the local contexts found in the training data. The local context for this module consists of three word forms to the left and right including their parts of speech. The second module also uses memory based learning, this time on keywords within the context of three sentences according to the method described by Ng and Lee (1996). The last module uses a rule learning algorithm on local contexts (as in the first module) and all content words within three sentences of the target word. Each module generates a word expert for each target word, and a sense assignment is made using majority and weighted voting.

2.6 Summary

We have described the beginnings of WSD, which included some system designs still in use today. Our focus shifted towards statistical knowledge–based WSD systems. We presented example systems for each of the three categories of knowledge–based systems: semantic approaches (Resnik, 1992), corpus–based approaches (Gale et al., 1992c) and dictionary–based approaches (Véronis and Ide, 1990). We motivated the need for baseline systems and described the most frequently used baselines. The chapter also introduces the Senseval exercise and presents the English lexical sample and English all words tasks from Senseval-2. The highest performing supervised systems in each category are described in some detail, with many traits (such as the types of features chosen) being common to all.

Chapter 3

Resources for WSD

3.1 Introduction

We present some resources often used for WSD, such as machine readable dictionaries and sense annotated corpora. The introduction of machine readable dictionaries leads us to a discussion of the notion of sense. We explain the difficulty of obtaining sense annotated corpora needed for supervised approaches to WSD, and investigate how the size of the training corpus affects the performance of a WSD system. We also describe related work on the automatic acquisition of sense annotated corpora.

3.2 Machine Readable Dictionaries

Most WSD systems choose senses for words from a pre-defined set of senses, which are enumerated in a machine readable dictionary. A number of these dictionaries exist: some are produced alongside creating a classical published dictionary, e.g., CIDE+ (Procter, 1995) or NODE (Pearsall, 1998); others are based on thesaural links, e.g., Roget's thesaurus (Roget, 1946). The most widely used machine readable dictionary in the WSD community is WordNet (Miller et al., 1990), which is based on psycholinguistic principles. This machine readable dictionary is perhaps most widely used in the community due to its free availability and continuing development. Before we present details of this dictionary (in Section 3.2.2), we discuss the much debated notion of sense.

3.2.1 Notion of Sense

Pustejovsky (1995) defines a sense enumeration lexicon (such as WordNet) as follows:

> A lexicon L is a *Sense Enumeration Lexicon* if and only if for every word w in L, having multiple senses $s_1, \ldots, s_n$ associated with that word, then the lexical entries expressing these senses are stored as $\{w_{s_1}, \ldots, w_{s_n}\}$.

Thus, a sense enumeration lexicon lists the senses of a word as you would find in a usual printed dictionary. It is distinguishing senses on the basis of some feature distinctions, and does not emphasise the difference between homonymy (where a word accidentally has two unrelated meanings) and other polysemy (where the meanings of a word are

related). This lack of distinction is the cause of the low performance of systems which return related, but wrong, senses that we discussed in Section 2.5, and it is the main motivation for probabilistic WSD systems.

Pustejovsky points out further deficiencies of the sense enumeration method of defining senses: for example, it does not account for sense extensions, where a word may be assuming different, but related, senses in new contexts (Copestake and Briscoe, 1995). Pustejovsky's solution to the problem is the *Generative Lexicon*, where words are only provided with a core set of senses, and rules are used to generate a larger set of word senses according to context. However, such a method makes an evaluation of WSD systems difficult as we are dealing with a potentially infinite number of senses.

Although not infinite, the number of senses in a sense enumeration lexicon differs from dictionary to dictionary (Kilgarriff, 1997). Even though it is possible to use tests to detect ambiguity (i.e., polysemy), for example the word *hard* is ambiguous as it has two unrelated antonyms *easy* and *soft*, such tests are never infallible (Zwicky and Sadock, 1975). As such, in many cases the decision whether to split a sense (divide a sense into multiple different meanings), or not, rests with the lexicographer. The WSD task may be more difficult when a fine-grained sense inventory is employed (one with many sense distinctions) instead of a coarse-grained one. Therefore we may observe large performance differences in WSD systems depending on the sense inventory.[1]

A more systematic method for classifying verb senses has been implemented by Levin (1993). She divided up verb senses according to the syntactic constructions they allow, e.g., the following are two different senses of the word *believe*:

1. *I believe it.* (noun phrase complement)

2. *I believe that he will leave.* (sentential complement)

However, her work has only been applied to a subset of verbs and therefore is not suitable for use in an all words WSD system. Despite the shortcomings of the sense enumeration lexicon, we use WordNet in our work due to the existence of corpora which have been annotated with its senses. This is primarily due to the general difficulty of obtaining sense annotated corpora, and the availability of corpora annotated with WordNet senses (see Section 3.3).

3.2.2 WordNet

WordNet (Miller et al., 1990) is a dictionary which, from its outset, was created as a machine readable dictionary for academic use. As it was designed for use by machines, the information it contains is not restricted to words with their definitions.[2]

Instead of an alphabetical listing of words, concepts in WordNet are grouped into synsets, where a synset of a word w is a set of w's synonyms. For example, the entry for *"a mixed breed dog"* contains not only *"mongrel"*, but also *"mutt"* and *"cur"*. The dictionary

[1] For example, the dictionary Hector was used in the SENSEVAL-1 evaluation competition, whereas the more fine-grained WordNet dictionary was used in SENSEVAL-2, and the average systems' performance dropped in the second evaluation competition (Kilgarriff, 2002).

[2] In this work, we use either WordNet version 1.7 (pre-release) or version 1.7.1. All words in these versions contain definitions.

is unusually structured as four separate part of speech groups (nouns, verbs, adjectives and adverbs).[3] Within each part of speech, synsets are listed in the most suitable method for the given part of speech. E.g., nouns are presented in an (IS-A) hierarchical structure (see Figure 3.1), whereas verbs are organized for example by entailment relations as well as a (much shallower) hierarchical structure. Although not used in our work, the entailment relation between X and Y holds if X can only hold if Y holds. For example, the verb *snore* entails the verb *sleep*.

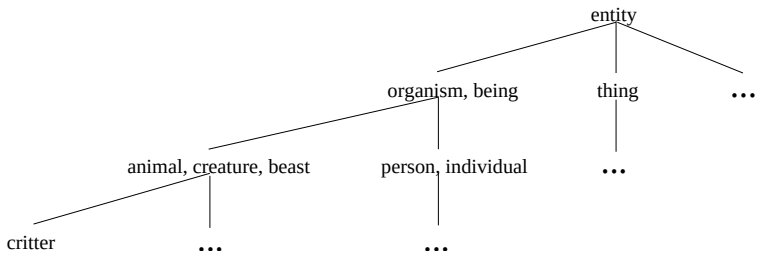

Figure 3.1: Hierarchical structure of WordNet

3.3 Annotated Corpora

WSD systems are either *unsupervised*, use no sense annotated data, or *supervised*. Supervised systems base their decisions on previously seen sense annotated examples, and usually outperform unsupervised systems (see e.g. Preiss and Yarowsky (2002) or Mihalcea and Chklowski (2004)).

However, obtaining manually sense annotated training data is generally difficult – it is a time-consuming task which requires skilled annotators to obtain a reasonable level of inter-annotator agreement, and is therefore expensive. Table 3.1 presents a list of existing manually sense annotated corpora. For each corpus, the table includes information about the sense inventory. We have automatically converted the DSO corpus[4] into WordNet 1.7.1 senses.[5] More detailed information about the converted corpora, including the corpus size and number of words annotated, is presented in Table 3.2.

3.4 How Much Data is Needed?

As supervised systems generally outperform unsupervised systems, the most important question is how much data will make them perform optimally. For example, the best

[3]We are not using the recently released WordNet 2.0, which contains links between different parts of speech.

[4]This corpus is available from the Linguistic Data Consortium, and was provided by Hwee Tou Ng of the Defence Science Organisation (DSO) of Singapore.

[5]Since WordNet is a very fine-grained dictionary, mappings to WordNet (where one sense from the original dictionary may map to more than one sense in WordNet) are usually possible, but they may not be very informative. E.g., sense s of word w in dictionary D may map to senses $\sigma_1, \ldots, \sigma_n$ in WordNet. We did not convert the Hector (SENSEVAL-1) and LDOCE (*interest*) corpora for this reason. We also did not convert the single word corpora from WordNet 1.5.

Corpus	Originators	Inventory
SEMCOR	Miller et al., 1990	WordNet 1.7.1
SENSEVAL-2	Kilgarriff, 2002, Palmer, 2002	WordNet 1.7 pre-release
DSO corpus	Ng and Lee, 1996	WordNet 1.5
line	Leacock et al., 1993	WordNet 1.5
hard	Leacock et al., 1998	WordNet 1.5
serve	Leacock et al., 1998	WordNet 1.5
SENSEVAL-1	Kilgarriff and Rosenzweig, 2000	Hector (Atkins, 1992–93)
interest	Bruce and Wiebe, 1994	LDOCE

Table 3.1: Existing sense annotated corpora

Corpus	Instances Labeled
SEMCOR	*brown1*: 106639 nouns, verbs, adjectives and adverbs *brown2*: 86000 nouns, verbs, adjectives and adverbs *brownv*: 41497 verbs
SENSEVAL-2	*English all words*: 2473 nouns, verbs, adjectives and adverbs *English lexical sample*: 12940 instances (15 adjectives, 29 nouns, 29 verbs)
DSO	192800 instances (121 nouns, 70 verbs)

Table 3.2: Corpus information

performing system in the SENSEVAL-2 English all words task, SMUaw (Mihalcea, 2002), is a supervised system which used a much larger training corpus than any other system participating in this task (using SEMCOR 1.6, WordNet example sentences, and GENCOR[6]). In this section, we extend the work of Mooney (1996) and investigate the hypothesis that the amount of training data directly affects the performance of a WSD system. As the F-measure of the SMUaw system is 69%, which only exceeds the most frequent sense baseline F-measure by 4%, the second aim of this investigation is to find the limits of supervised WSD systems. For example, this could be represented by a performance plateau when graphing performance against the amount of training data, such as in the work of Banko and Brill (2001) whose algorithm reached a plateau when applying machine learning to confusion set disambiguation with up to 1-billion words of training data.

Yarowsky and Florian (2002) investigate the performance sensitivity to the size of the training corpus used for a number of supervised algorithms. However, their work uses the SENSEVAL-2 test data for the experiment and they only enforce the need to have one example per sense, leading to fairly small variation in the size of the training corpus. In our investigation, we use our modular probabilistic WSD system, and train on varying amounts of the *line* corpus (see Table 3.1, and description below).

[6]GENCOR is the author's automatically generated sense annotated corpus (Mihalcea and Moldovan, 1999). It is not publicly available.

3.4.1 WSD System

Although the results of this experiment are restricted as we are dealing with a one word corpus with a six senses which are distributed in a certain way (and so the results may not carry over to all other words), the experiment is crucial in illustrating the necessary limits of current supervised systems. These limitations therefore motivate the need to incorporate unsupervised components in WSD systems.

For the purposes of explaining this investigation, we present a brief overview of our probabilistic WSD system – a more thorough description of the system can be found in Chapters 6 and 5. The WSD system uses Bayesian statistics to combine twenty modules, of which only two are unsupervised. The remaining 18 all require training data.

3.4.2 Evaluation Corpus

The investigation is carried out using the *line* corpus. This corpus contains 4148 instances of the word *line*, labeled with one of the six senses which were considered by the authors (Leacock et al., 1993). This represents the largest number of manually annotated instances of a single word. Table 3.3 shows each of the senses of *line* with two pairs of frequency and corresponding rank value – the first pair is obtained from the *line* corpus itself, the second is taken from WordNet 1.5. The WordNet frequencies are based on the associated SEMCOR corpus, which comprises 352 texts taken from the English Brown corpus, and was manually annotated by lexicographers. It is interesting to note that the frequency ranking of the senses obtained from the *line* corpus is very different to that from WordNet. This leads us to do two types of experiment: the first using the entire corpus but starting with various frequency distribution priors, and the second based on a randomly generated balanced part of the *line* corpus.

Sense	Example sentence	Line corpus		WordNet 1.5	
		Freq	Rank	Freq	Rank
product	*A nice line of shoes.*	2217	1	1	5
phone	*Please, stay on the line.*	429	2	2	3
text	*The letter consisted of three short lines.*	404	3	9	2
division	*There is a narrow line between sanity and insanity.*	376	4	0	6
cord	*A washing line.*	373	5	1	4
formation	*The line stretched clear around the corner.*	349	6	16	1

Table 3.3: Frequency and rank information for the *line* corpus

3.4.3 Experiment 1

Our first experiment uses the complete *line* corpus, which we split into a test corpus, a development corpus (used for acquiring smoothing values), and a training corpus. The size of the test corpus is 174 instances as is the size of the development corpus. The remainder

of the *line* corpus forms the training corpus, which is further divided up into 18 (unequal) segments. This allows us to observe performance variation with an increasing training corpus. The segments are combined to create training corpora of varying sizes, the n-th training corpus will be of size $\sum_{i=1}^{n} 20i$; i.e., there are corpora of the following sizes: 20 instances, 60 instances, 120 instances, 200 instances etc. To create the 60 instance training corpus, we add 40 new instances to the initial 20 instance training corpus; this 60 instance training corpus is increased by another 60 instances to create the 120 instance corpus and so on. It is important that the training corpus is created in this incremental fashion, as the larger corpora will contain the training instances in the smaller corpora (and therefore performances can be directly compared). The experiment using the 18 different training corpora is repeated 10 times (with randomly generated test, development and training corpora), to create reliable average performances.

The F-measure for the most frequent sense baseline using the *line* corpus frequencies is 50.1% (precision is 53.2%, recall 47.3%). This F-measure is obtained by averaging over 18 randomly chosen test corpora of 174 instances. If the corpus frequencies from WordNet 1.5 are used instead, the F-measure for this baseline is only 8.3% (precision is 8.8%). The performance difference of the two most frequent sense baselines is clearly due to the different distribution of senses in SEMCOR compared to the *line* corpus seen in Table 3.3.

Based on underlying frequency distributions, we create three versions of our WSD system for this experiment: the first based on the WordNet 1.5 frequencies, second taking its frequency information from the *line* corpus and the last being based on equal probabilities for each sense. The performance graph of the WSD system (F-measure) against size of training data can be found in Figure 3.2. *WordNet* denotes the algorithm starting with WordNet 1.5 frequencies, *line* starts with the frequencies taken from the *line* corpus, and *uniform* starts with a uniform distribution on senses (each sense having a $\frac{1}{6}$ probability).

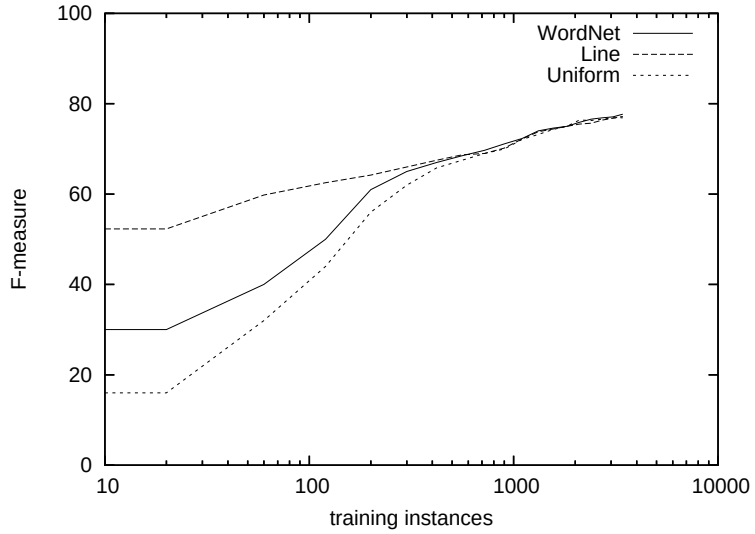

Figure 3.2: F-measure with WordNet, *line* and uniform frequencies

3.4.4 Experiment 2

For the second experiment, we reduced the *line* corpus so all senses were represented equally. The occurrences were randomly chosen, each sense being represented 349 times and therefore only 2094 instances are used. To allow our results to be comparable to Mooney's (1996), the corpus was split into 1200 training instances and 894 test cases. Mooney compared the performance of seven different learning algorithms for WSD on the DSO corpus. He also plotted performance against training data graphs for each system, indicating a possible plateau (although the number of points on the graph were limited) and reaching a maximum precision of around 72%.

We split our training corpus further into 10 (unequal) segments to give incremental training corpora containing 20 instances, 60 instances, 120 instances, ... (as described in Experiment 1) and leaving 100 instances for development. This experiment using the 10 different corpora was repeated 10 times (again with randomly generated test, development and training corpora). The performance graph of the WSD system (average F-measure) against the size of training data can be found in Figure 3.3.

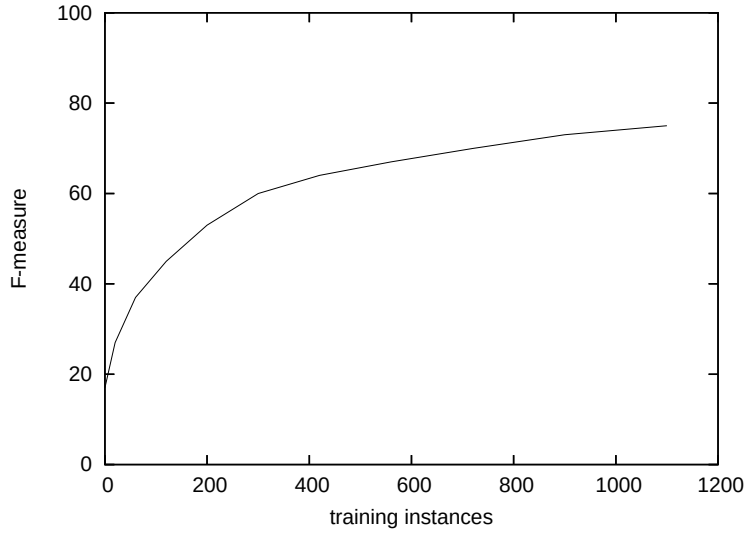

Figure 3.3: F-measure with balanced training data

3.4.5 Discussion

Most importantly, we can observe that the gradient on both performance graphs becomes much lower after its initial high, indicating that there is probably a performance plateau for the system used. The graph shape seen in Figure 3.3 is similar to that presented in the work of Mooney (1996), whose WSD algorithm consisted of various machine learning techniques. This suggests that the shape of the graph may be typical of pattern-based supervised approaches to WSD.

Secondly, Figure 3.2 indicates that using around 200 instances dramatically reduces the performance gap between the different frequency information used by the WSD system. At this point, the performance of our WSD system based on the uniform probabilities is

56% F-measure, using the WordNet 1.5 probabilities it is 58% F-measure and if the *line* corpus probabilities are used the F-measure is 62%. However, around 500 instances are needed to compensate fully for wrong prior frequency information.

It is important to put the number of training instances into context: for example, the average polysemy over all words with more than one sense in WordNet 1.7.1 is 3. Assuming SEMCOR is used to train an all words system, words with polysemy 3 will only have 5 training instances. Now consider the SENSEVAL-2 English lexical sample task where training data was provided. The number of instances tagged for each word in the English lexical sample task was $75 + 15n$ where n is the number of senses the word has within the chosen part of speech (Kilgarrif, 2002; Palmer et al., 2002)). Two thirds of the total annotated data for each word was used for training (see Figure 3.4 for a diagrammatic representation of the average number of training instances against the number of senses). Even if we consider the word *begin* with 11 senses, which has by far the highest number of annotated training instances (557, the next being the word *day* with 289 training instances), this is still much lower than the 4000 instances tagged for the six senses of the word *line* which we have investigated. As 500 training instances were needed to compensate for errors in the prior for a 6 sense word with distinct senses, this implies that the success of a SENSEVAL system trained only on the data provided depends largely on its choice of prior.

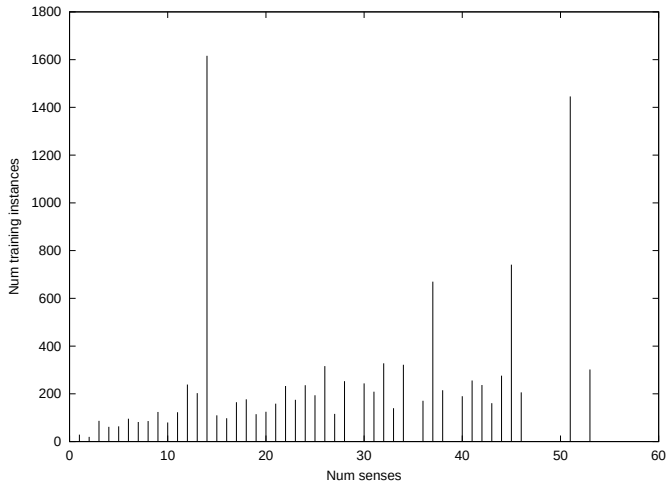

Figure 3.4: Number of training instances for the English all words task

It is interesting to see that even with the high number of training instances used, the performance of the WSD system does not appear to improve much above an F-measure of 75%. To investigate whether this is a limitation of the system or the training corpus, we used the same data to train and test on. This gives us an approximate upper bound for the system with F-measure of 92% (precision 97.9%, recall 87.4%),[7] which indicates that even 4000 instances for a six sense word is not enough to achieve the highest possible performance with the current system.[8] A possible technique for overcoming the lack of

[7]As our system is probabilistic, it is theoretically possible that in certain circumstances the system would perform better than this upper bound.

[8]Note that these conclusions are drawn from results based on one noun which has a certain distribution

training data is automatic training corpus acquisition, and we summarize related work on this topic in the next section.

3.5 Automatic Training Corpus Acquisition

We now present related work on automatic methods of obtaining sense tagged data. Such approaches are often noisy (introducing wrongly annotated examples), but they remove the need for skilled annotators.

3.5.1 Related Work

Bootstrapping methods, where the system starts with a number of manually tagged examples for each word, and iteratively uses classifiers to sense tag new words (which are added to the tagged examples), have proved to be quite successful (Hearst, 1991; Yarowsky, 1995; Mihalcea, 2004). However, this method does require a number of manually sense tagged examples for every word (and therefore every sense) we are interested in.

A large amount of effort has been spent on making use of bilingual corpora. Gale et al. (1992c) used an aligned parallel corpus to sense tag words which have different translations in their various senses. Such as, for example, the financial sense of "*bank*" translates to French as "*banque*", whereas the riverside sense of the word is "*rive*". A possible drawback of this approach is that ambiguities may be preserved in the two languages (Gale et al. used French and English, a pair of quite related languages). A more serious failing comes from the lack of aligned parallel corpora.

Possible approaches to solving the second problem were suggested by Dagan and Itai (1994), who used two monolingual corpora with a bilingual dictionary: they use a bilingual dictionary to generate possible translations, and rule out incorrect ones by checking the frequencies of syntactic relations involving the individual words within the translation in the target language. However, the first problem remains.

An automatic method for creating sense tagged data based on a dictionary (and therefore no additional sense tagging data) was initially suggested by Leacock et al. (1998), who employed the WordNet synonymy relation to identify instances containing monosemous words, which could be used as instances for their (now sense tagged) ambiguous synonyms.[9] For example, in the sentence

> A *chime* is a musical instrument.

they would replace the occurrence of *chime* with its synonym *gong* to yield a sense tagged training instance of the word *gong*

> A *gong* is a musical instrument.

However, the number of monosemous words is limited and so a rather skewed number of examples was being obtained. The method was therefore refined by Mihalcea and Moldovan (1999) who used web queries based on dictionary definitions to retrieve examples. Searches were made not only for monosemous words (corresponding roughly to

on the six senses chosen.

[9] Leacock et al. also used the hyponymy relation.

the Leacock et al. method), but also for definition words from WordNet. They used this method to create a corpus of around 160,000 sense tagged words (GENCOR). For more details on the high performing SMU systems which made use of GENCOR, see Section 2.5.1.

3.6 Summary

We have discussed the notion of senses and presented machine readable dictionaries with their annotated corpora, which we will use in Chapter 6 to train our system. With an investigation of the effect of the amount of training data on the performance of WSD, we showed that the amount of training data usually used is very small compared to that needed. We also presented related work on automatic acquisition of sense annotated corpora.

Chapter 4

Decision Trees for WSD

4.1 Introduction

To motivate the modular approach taken in this thesis, we compare the WSD systems submitted for the English all words task in SENSEVAL-2. We give several performance measures for the systems, and analyze correlations between system performance and word features. A decision tree learning algorithm is employed to discover the situations in which systems perform particularly well, and the resulting decision tree is examined. We investigate using a decision tree based on the SENSEVAL systems to (i) filter out senses unlikely to be correct, and to (ii) combine WSD systems.

4.2 Analysis of System Results

4.2.1 Feature Correlations

Systems rely on various resources to perform disambiguation, such as training examples in SEMCOR or definitions in WordNet. This section illustrates most clearly the novelty in our work: WSD comparison studies such as Yarowsky and Florian (2002) have re-implemented a small number of algorithms and focused mainly on the effect of varying internal parameters (e.g. the window size used by the algorithm). This study uses the answers of 21 existing WSD systems used for the English all words task in SENSEVAL-2 to investigate the relationship between system precision and the most frequently used features (see system descriptions in Table 2.7). Positive correlations could reveal how to combine systems in a future WSD system to optimize performance.

As there was no training data available for this task, a number of the supervised systems used SEMCOR as their training corpus. Table 2.6 suggests that the most frequent sense baseline performs better than all but two systems.[1] It uses the WordNet 1.7 `cntlist` (frequency information file) to provide the ranking of senses, and this frequency information was obtained from SEMCOR.[2] Indeed, a correlation graph shows a relation

[1] However, this baseline has access to perfect part of speech information about the word, and also has perfect knowledge of multi-words in the text.

[2] As an official WordNet 1.6 to WordNet 1.7 mapping has not yet been released, we have used SMU's mapping from WordNet 1.6 to WordNet 1.7 for SEMCOR to find out whether most of the training examples in SEMCOR could be used by a supervised system. The number of wrongly tagged instances was very

exists for supervised systems between precision on a word and number of occurrences in SEMCOR (correlation values around 0.1 for supervised systems).

However, the distribution of senses in training data (and indeed occurrences in text) tends to be skewed towards the most frequent sense (e.g. Kilgarriff and Rosenzweig (2000)). We therefore investigate the precision of the systems on words where the most frequent sense is not correct; these words form 32% of the corpus (790 instances). The precision of the systems on these words varies between 11% (DIMAP) and 31% (SMUaw) with an average of 18%, and random achieving 17%. The SMU system Mihalcea and Moldovan (2002) for the English all words task uses "patterns" (local contexts) learnt from SEMCOR, GENCOR and WordNet definitions. We note that the training corpus employed by the SMUaw system is larger than that used by other systems. This suggests that the size of their corpus may be beginning to balance out the skewed distribution of occurrences of word senses in texts to provide enough training examples to learn patterns from even the less frequent senses.

The SMU system is not the only one to use the WordNet definitions; these are also used for disambiguation by the UNED systems and by the DIMAP system. Therefore, we next investigate the correlation between definition length and precision: if a definition is used for overlap and the words in the definition are relevant, then the longer the definition, the greater the chance of an overlap occurring. Thus, we would expect a positive correlation between the precision of systems using the definitions and definition length. WordNet 1.7 includes a gloss (a definition and possibly some example sentences) for every word in the dictionary.[3] The correlation graph can be seen in Figure 4.1.

The system with the highest correlation is Sussex-sel, which is based on selectional preferences and surprisingly does not make any use of the WordNet definitions. However, these results must be placed in their proper context. The correlation of definition length with polysemy is -0.78, which means that the longer the definition, the less polysemous the words are. Therefore, if the Sussex-sel system attempted words which were below average in polysemy, it would be expected to have a higher correlation with definition length. This is true: the average polysemy within the correct part of speech of the words attempted by the Sussex-sel system is 3.36 (which is lower than the WordNet average of 5.24). This also explains why the most frequent sense baseline is the third most correlated.

This leads us to investigate the correlation of polysemy with precision. We would naively expect polysemy to be inversely correlated to precision – the more senses a word has, the more difficult it may be to disambiguate.[4] For each system we compute the precision on words of all polysemies separately. These are then correlated to produce a correlation value for each system, presented in Figure 4.2. Remarkably, the IIT systems correlate positively with polysemy (the low recall of the IIT systems is not due to their choice of words to disambiguate but a bug which meant that only answers for the initial 20% of the corpus were submitted). These systems use WordNet examples to score the similarity of the word's local context in the text to the example. The relative success of IIT on more polysemous words therefore supports the need to take some context into account

small and we therefore concluded that most supervised systems could have used SEMCOR in full.

[3]For this experiment, we remove the example sentences (since they frequently use only unrelated words), then lemmatize and stoplist the remaining definition. Definition length is then the number of remaining words.

[4]Consider the difference between a multiple choice test with three answers per question, versus one with fifteen answers per question.

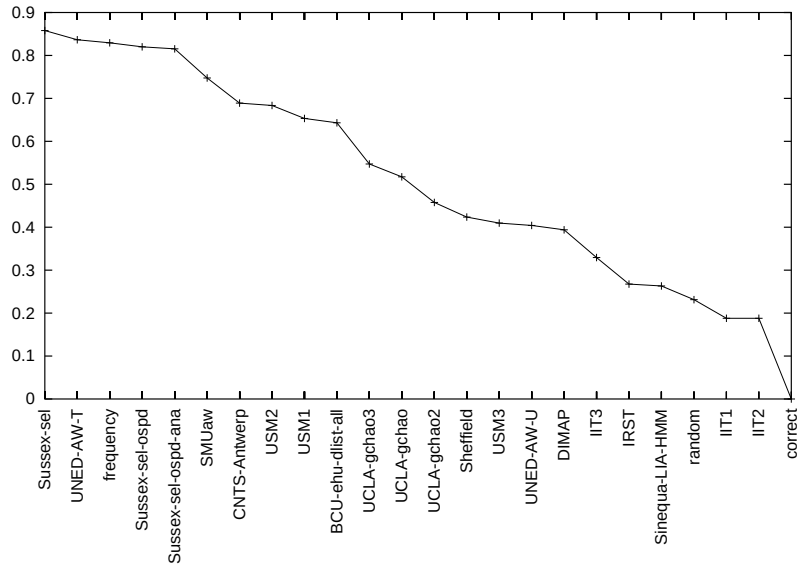

Figure 4.1: Correlation of WordNet definition length with system precision

in disambiguation. The remaining systems do not appear to have a high correlation, supporting the suggestion of Kilgarriff and Rosenzweig (2000) that polysemy is not an ideal measure of difficulty.

Polysemy	Better distribution	Worse distribution	Confidence
2	WordNet class	WordNet class	99.5%
3	of the word	of the word	99.5%
4	**does not**	**contains**	95.0%
5	**contain any**	**multiple**	99.5%
6	**other sense** of	**senses** of the	99.5%
8	the word.	word.	90.0%

Table 4.1: *t*-test comparing precision on words with multiple senses in WN classes

We also examined the precision of systems on words where the WordNet class of the correct sense did not contain any other senses of the word. All words in WordNet belong to one of 44 classes: adverbs are not split into classes; there is a class for descriptive adjectives and relational adjectives; and nouns and verbs are classed according to their semantic fields (Miller et al., 1990). We found the precision of each system on words where the correct sense was in a WordNet class not shared with other senses (isolated), and on words where both the correct sense and an incorrect sense were in the same WordNet class (shared). These were further split up according to the degree of polysemy. The results in Table 4.1 show the results of *t*-tests of precision variation between the two distributions for each level of polysemy.[5] These clearly indicate improved precision where the correct WordNet class did not contain other candidate senses regardless of the degree of polysemy.

[5]The *t*-test is only performed for degrees of polysemy containing at least 5 significant systems, where a significant system is one that attempts at least 20 words in both categories.

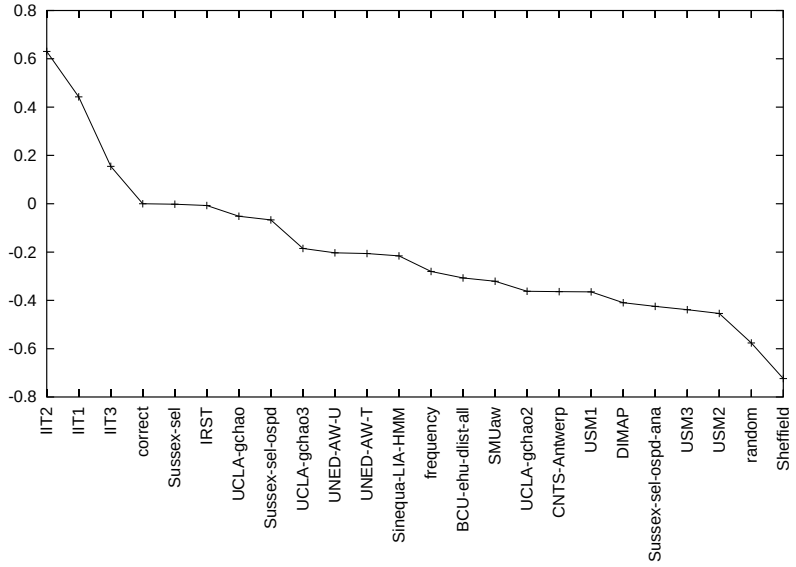

Figure 4.2: Correlation of polysemy with system precision

4.2.2 Linguistic Analysis

This section presents a short analysis of the words which were mislabeled by at least 20 (out of the 21) systems, subject to the constraint that the words occurred in the corpus at least three times.[6]

The most frequently mislabeled nouns were *form, one, change, bang, service, loss,* and *development.* These words were often mislabeled with a 'related' sense; for example, the word *change* occurs three times in the corpus in the change%1:19:00:: sense: *the result of alteration or modification.* Many systems chose instead change%1:11:00, which is defined as *an event that occurs when something passes from one state or phase to another.* However, the synonyms of change%1:11:00 are *alteration* and *modification*, and thus for a definition and synonym overlap based system the two senses are quite likely to have similar overlap scores.

In the case of verbs, the most frequently mislabeled ones were *lead, turn, make, find, miss, say, involve, think, show,* and *develop.* These verbs have a number of closely semantically related senses, which systems often confused.[7] For example, lead%2:42:12:: (*tend to or result in*) was often confused by systems with lead%2:42:04::, which is defined as *result in.*

The most frequently mislabeled adjectives were *common, rare, defective,* and *simple.* All of these were wrongly annotated when used in the satellite sense – the systems preferred to select a non-satellite adjective sense. This is not surprising as the satellite synset for a satellite is designed to represent a similar concept to the head adjective synset. This finding indicates that a promising approach to improving the performance of systems on adjectives is to properly deal with satellites.

[6]We are considering each word only within one part of speech, i.e., we are assuming a perfect part of speech tagger.

[7]Senses which are in the same WordNet class are thought to be semantically related.

4.3 WSD System Specializations

4.3.1 Decision Tree Learning

The correlations observed in Section 4.2.1 support the hypothesis that different systems may be better at disambiguating different words, which we discussed in Section 5.2. However, in this section, our goal is not to create a WSD system, instead we aim to analyze the types of words for which each system is suited. Types of words are defined by our choice of features which we correlate, e.g. words with high frequency, words with polysemy equal to two. To carry out the analysis, we chose ten features $F_1, \ldots, F_{10}$ (see Table 4.2) to use in decision tree learning. We now briefly justify our choice of features:

F_1 – The **position in sentence** will affect bigram or trigram information (potentially none present as no preceding words). E.g. BCU-ehu-dlist-all.

F_2 – For example, IIT3 uses already chosen senses (for preceding words) to disambiguate the target word. In this case, the **number of words attempted in a sentence** may help indicate how confident the system can be about its answer.

F_3 and F_4 – **Overall polysemy** and **polysemy within the given part of speech** affects all systems. In the case where the target word is monosemous in the correct PoS (but is polysemous over all), F_4 will tell us about the accuracy of the tagger used by the systems.

F_5 – A system which relies on the WordNet hierarchy (which is more detailed for nouns than for other PoS), may perform worse on particular **parts of speech**. E.g. Sussex-sel.

F_6 – The performance of a system which uses definition overlap (e.g. DIMAP) may be affected by the **length of the definitions**.

F_7 – IRST and Sinequa-LIA-HMM systems make use of the WordNet semantic classes. It may therefore be useful to know **how many other senses of the target word are in the class of the chosen sense**.

F_8 – The **system name.** A particular system may be very accurate on nouns but not verbs, so a sense assignment for a noun could always be made based on this system.

F_9 – The **number of training examples for the chosen sense** may influence the performance of supervised systems, also see Section 3.4.

F_{10} – The **frequency rank of the chosen sense** may affect our confidence in the system's answer, if the system backs off to the most frequent sense, e.g. Sheffield.

Decision tree learning thus appears well suited to our task: our target function is discrete valued (the annotation is either correct or incorrect) and the instances can be described as attribute-value pairs in terms of our features. We are also expecting the training data to be noisy and decision tree learning is usually able to cope with this. For an introduction to decision tree learning, see e.g., Mitchell (1997).

No.	Feature	Values
F_1	`position_in_sentence`	[0, 1]
F_2	`words_attempted_in_sent[ence]`	real
F_3	`polysemy_overall`	real
F_4	`polysemy_within_attempted_pos`	real
F_5	`answer_pos`	{Noun, Verb, Adj, Adv}
F_6	`definition_length`	real
F_7	`word_difficulty [within class]`	real
F_8	`system`	system names
F_9	`num_training_egs`	real
F_{10}	`frequency_rank`	real

Table 4.2: Features in decision tree

We use the WEKA (Witten and Frank, 2000)[8] implementation of the C4.5 decision tree learning algorithm (Quinlan, 1993), to create a decision tree of system results and word features. The training instances consist of the features in Table 4.2 relative to the system answer. Training instances are classified as positive or negative depending on whether the system disambiguated them correctly. See Table 4.3 for an example of the input to the decision tree learner of the test instance d00.s00.t01. Note that a system may appear more than once per test instance, if it submitted more than one answer.

4.3.2 Systems Specializations

The C4.5 algorithm constructs a decision tree top-down, by selecting the attribute that best classifies the local training examples at each node. When faced with a testing example, the tree can be followed from the top-most node to a classification. An excerpt from the system decision tree we obtain is displayed in Figure 4.3.

The system decision tree is a good fit to the data: the percentage of correctly classified instances on the training data is 94.59% and the accuracy of a 10-fold cross validation is 91.83%.[9] This internal cross-validation shows us that the structure of the tree is robust.

The aim of creating the decision tree is to determine combinations of features that increase or decrease the performance of particular systems. Our goal would therefore be best served if the decision tree split on the SENSEVAL systems at a high-level, allowing for many individual variations beneath each system. For example, we would expect to see a split on `num_training_egs` with supervised systems or a split on definition length with systems which used definitions.

Monosemous words, which account for 5464 out of the total 34107 classifications, will obviously not require a system to be disambiguated. It is surprising to discover that the decision tree makes a further 21604 classifications (out of 34107) without splitting on the systems at all. Each of the branches leading up to the 27068 (21604 + 5464) classifications defines a type of words (e.g. monosemous words for the `polysemy_overall` ≤ 1 branch leading up to 5464 classifications). Our result means that for these types of words, any system which submits an answer is 'equally good' as any other system which submitted

[8] Available from `http://www.cs.waikato.ac.nz/~ml/weka/`.

[9] Note that these values do not represent the accuracy of a combined WSD system. They only show how well the decision tree fits the data.

F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	class
0.00	4	4	1	4	6	0	BCU-ehu-dlist-all	50	1	no
0.60	11	4	1	4	6	0	frequency	50	1	no
0.50	11	4	1	4	6	0	UCLA-gchao	50	1	no
0.90	11	4	1	4	6	0	UCLA-gchao2	50	1	no
0.90	11	4	1	4	6	0	UCLA-gchao3	50	1	no
0.00	11	4	1	4	6	0	UNED-AW-T	50	1	no
0.90	11	4	1	4	6	0	UNED-AW-U	50	1	no
0.40	11	4	1	4	6	0	USM1	50	1	no
0.40	11	4	1	4	4	0	USM2	16	2	no
0.40	11	4	1	4	6	0	USM3	8	3	yes
0.00	11	4	1	4	6	0	IIT1	50	1	no
0.00	11	4	1	4	6	0	IIT1	8	3	yes
0.00	11	4	1	4	6	0	IIT2	50	1	no
0.00	11	4	1	4	6	0	IIT3	50	1	no
0.00	11	4	1	4	6	0	IIT3	8	3	yes
0.40	11	4	1	4	6	0	CNTS-Antwerp	50	1	no
0.70	11	4	1	4	6	0	DIMAP	50	1	no
0.40	11	4	1	4	4	0	Sinequa-LIA-HMM	16	2	no
0.20	6	4	1	4	4	0	IRST	16	2	no
0.40	11	4	1	4	6	0	SMUaw	50	1	no
0.25	5	4	1	4	6	0	Sheffield	50	1	no

Table 4.3: Example of training input to decision tree learning algorithm

an answer. Thus for these types of words, we don't need to employ twenty-two different systems, but we only need one which will always provide an answer. This criterion is satisfied by the most frequent baseline system. Note that the proportion of classification made without using a particular system is 79%, this supports the 80/20 rule of time management (Pedersen, 2001), which suggests that 20% of effort accounts for 80% of results.

The two subtrees beneath the splits on the SENSEVAL systems account for 7039 classifications. The initial decisions leading up to the 'system' split are:

1. (frequency_rank ≤ 1) $\wedge$ ($2 <$ polysemy_overall ≤ 4) $\wedge$
   ($2 <$ definition_length ≤ 9) $\wedge$ (polysemy_within_attempted_pos ≤ 2) $\wedge$
   (num_training_egs ≤ 3)

2. ($1 <$ frequency_rank ≤ 3) $\wedge$ (polysemy_overall > 1) $\wedge$
   (definition_length ≤ 15)

The first case deals with the situation when the suggested sense is the highest frequency sense (of a word which is not highly polysemous). In this case, the systems can be trusted in their positive or negative answers and quickly lead to a final decision (there is at most one further split in the decision tree). But this subtree only accounts for 156 classifications (out of 7039).

A decision is not reached quite as quickly in the second case. However, it can be seen that classifications are independent of SENSEVAL systems whenever the frequency rank of a suggested sense exceeds 3.

It is interesting to note that the number of further splits varies from system to system, as do the features involved. This is where we hoped to gain useful insight into the

```
polysemy_overall <= 1: yes (5464.0)
polysemy_overall > 1
|   frequency_rank <= 1
|   |   definition_length <= 13
|   |   |   polysemy_overall <= 2
|   |   |   |   num_training_egs <= 4
|   |   |   |   |   word_difficulty <= 0
|   |   |   |   |   |   num_training_egs > 2
|   |   |   |   |   |   |   definition_length <= 2
|   |   |   |   |   |   |   |   words_attempted_in_sent <= 10: yes (17.0)
|   |   |   |   |   |   |   |   words_attempted_in_sent > 10: no (6.0)
|   |   ..........
|   frequency_rank > 1
|   |   definition_length <= 15
|   |   |   frequency_rank <= 3
|   |   |   |   system = BCU-ehu-dlist-all
|   |   |   |   |   ..........
|   |   |   |   system = USM1
|   |   |   |   |   polysemy_overall <= 3
|   |   |   |   |   |   word_difficulty <= 1
|   |   |   |   |   |   |   words_attempted_in_sent <= 19: no (131.0/26.0)
|   |   |   |   |   |   |   words_attempted_in_sent > 19
|   |   |   |   |   |   |   |   num_training_egs <= 5: yes (6.0/1.0)
|   |   |   |   |   |   |   |   num_training_egs > 5: no (2.0)
|   |   |   |   |   ..........
|   |   |   |   system = USM2
|   |   |   |   |   ..........
```

Figure 4.3: An excerpt from the decision tree

situations in which some or all of the systems perform well. For example, looking again at the excerpt in Figure 4.3, beneath the USM1 system the first split is whether the overall polysemy of the word is less than or equal to 3, suggesting that performance might be significantly different in each of these cases.[10] However, there are unfortunate correlations between the features that make the decision tree very hard to interpret. Note in Figure 4.3, beneath USM1 there is a split on the number of training examples in SEMCOR. However, the USM1 system is unsupervised and does not perform training, so this split must represent some hidden dependence between the features that happens to make this split the most informative. Many other examples of this phenomenon occur underneath the main system split, and so unfortunately we cannot draw many conclusions about the system specializations.

But there are still two striking and surprising features of the resulting decision tree:

1. For the majority of words, the individual systems are not split upon to make an optimal decision.

2. The systems are split upon at just two different points in the decision tree.

From this comparison, we conclude that in building a future WSD system we should

[10] In fact, the performance of USM1 is 43% when the polysemy is less than or equal to 3 (but greater than 1), and 16% otherwise.

focus mainly on identifying the words which cannot be decided using a baseline system and optimizing the new system for these difficult words.

4.4 Systems

A decision tree based on the inputs of a number of WSD systems, as described in Section 4.3, could be employed in new WSD systems. We present two example applications:

1. A decision tree used as a component of a WSD system, which rules out certain senses from consideration.

2. A full system which, based on the output of some of the systems, chooses a sense for each word.

In the following two subsections we investigate these applications.

4.4.1 Filter System

Instead of using the SENSEVAL systems to produce a sense for each word in a corpus, we investigate their suitability as filters. For each system, we learn a decision tree from 90% of the system's answers and find out how frequently the decision tree classifies the correct sense as incorrect (i.e. the correct sense is filtered out) on the remaining 10% of data. This is motivated by the assumption that although a word may have a number of senses, a system may only be deciding between a few of them. In that case, it would be useful to be able to reliably rule out some of these. For example, our WSD system may immediately rule out the two inanimate senses of the word *dog* (6 senses in WordNet). If we can reliably use the SENSEVAL systems to say that *dog* cannot have the "You lucky dog" sense, our WSD system is only deciding between 3 remaining senses.

For this task, we make use of the decision tree created in Section 4.3.2, which also has the following branch:

$$(2 < \text{polysemy\_overall} \leq 4) \wedge (\text{frequency\_rank} \leq 1) \wedge$$
$$(2 < \text{definition\_length} \leq 9) \wedge (\text{polysemy\_within\_attempted\_pos} \leq 2) \wedge$$
$$(\text{num\_training\_egs} \leq 3) \wedge (\text{system} = \text{USM1}) \Rightarrow \text{no}$$

This means that if certain conditions hold about a sense suggested by the USM1 system, the decision tree predicts that this is an incorrect annotation. The percentage of misclassifications (filtering out the correct sense) for each system is presented in Figure 4.4 (systems are ordered according to their F-measure in this figure).[11] We can see that some systems appear to be better suited to the filtering task than others: the Sussex-sel system (unsupervised, based on selectional preference information) only rules out 2.3% of the correct senses, whereas the BCU-ehu-dlist-all system (supervised, using decision lists) rules out 11.9%. However, the BCU-ehu-dlist-all system also attempted many harder words (both in terms of polysemy and in terms of our difficulty, defined in Section 4.2.1) than the Sussex-sel system, which may explain some of the difference.

[11]Note that in most cases when the systems rule out a sense, it is not the correct one and thus the percentage of misclassifications is not equal to precision.

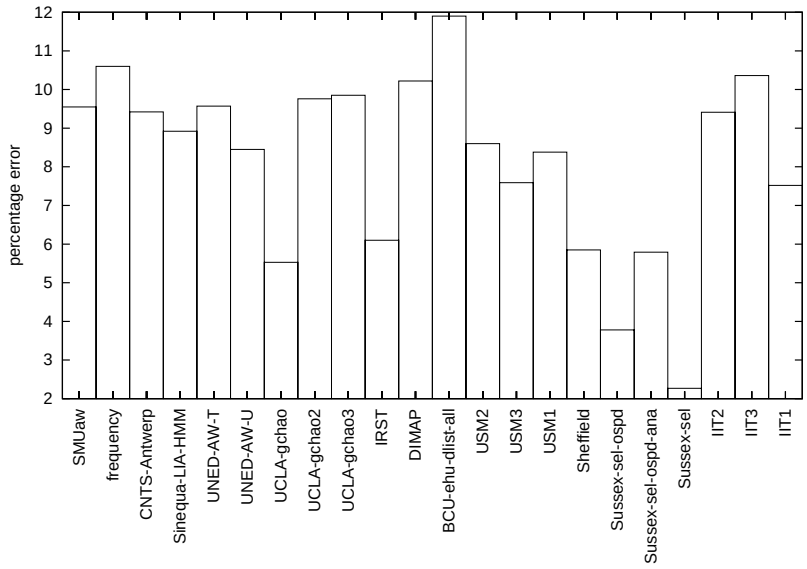

Figure 4.4: Misclassifications in Filtering

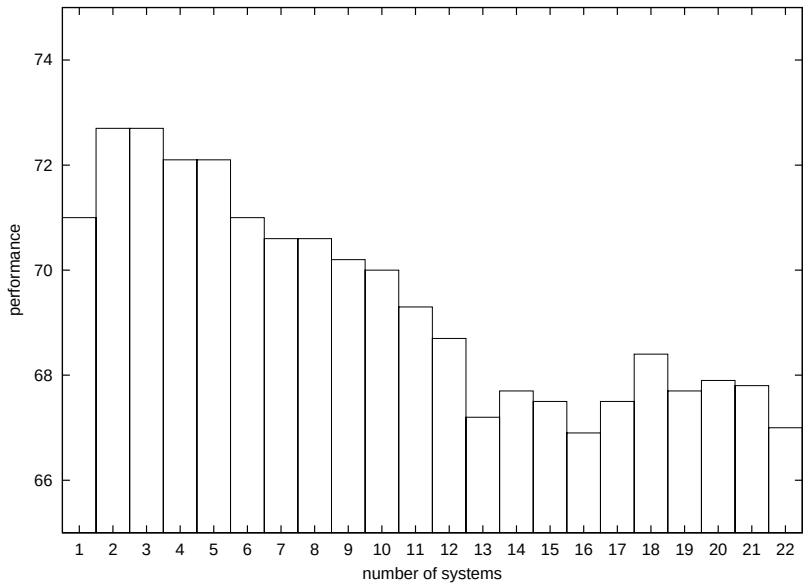

Figure 4.5: Combined Systems

4.4.2 Combined Systems

For each word, we group together the answers given by a number of the SENSEVAL systems. The decision tree is created from the answers to 90% of all words. For each word to be classified, we pass a number of system answers to the decision tree. As part of its classification, the decision tree produces a confidence in a "yes" or a "no" answer. The confidences in the answers are then ranked and the sense with the highest confidence is chosen.[12]

We ranked the systems according to their F-measure (producing the list: SMUaw, frequency, CNTS-Antwerp, Sinequa-LIA-HMM, UNED-AW-T, UNED-AW-U, UCLA-gchao, UCLA-gchao2, UCLA-gchao3, IRST, DIMAP, BCU-ehu-dlist-all, USM2, USM3, USM1, Sheffield, Sussex-sel-ospd, Sussex-sel-ospd-ana, Sussex-sel, IIT2, IIT3, IIT1). A number (between 1 and 22) of top-ranked systems was then combined to create a WSD system, which was tested using the method above. The results of the 10 fold cross validation are presented in Figure 4.5. In this table, "number of systems = 1" represents the performance of the decision tree when it is only trained and tested on the SMUaw system, "number of systems = 2" represents the performance when trained and tested on both SMUaw and frequency, etc.

An excerpt from a (one-tailed) t-test comparison between all pairs of combined systems is presented in Table 4.4. The full matrix tells us which combined systems are significantly better than others. Due to space constraints, we only present the section of the table which shows that the combined systems significantly outperform the best SENSEVAL-2 system. This table is to be interpreted as follows:

	1	2	3	...
2	99.0	*	–	...

shows that a combination of two systems outperforms one system with a confidence of 99%. However, the combination of two systems does not outperform three systems (indicated by "–"). Table 4.4 shows that combining the top 2, 3, 4, or 5 outperforms the current best system with varying degrees of confidence. Combining more than the top five systems does not lead to a more accurate system. From our investigation, it seems the that best system results from combining the SMU, the most frequent and the CNTS-Antwerp

[12]In order to be able to truly choose the highest confidence, we convert 95% confidence in "no" into a 5% confidence in "yes", so all confidences are presented in terms of "yes".

Combined systems	Combined systems				
	1	2	3	4	5
1	*	–	–	–	–
2	99.0	*	–	80.0	70.0
3	99.5	50.0	*	85.0	75.0
4	99.5	–	–	*	–
5	97.5	–	–	50.0	*
6	–	–	–	–	–

Table 4.4: Excerpt from the Performance Comparison of Combination Systems

systems. Although both SMUaw and CNTS-Antwerp back-off to the most frequent sense, they can be seen to be potentially complementary. The SMUaw system learns rules from local contexts. The CNTS-Antwerp system also includes a word expert which considers keywords from a context of 3 sentences.

It is interesting to note that combining the SMU system with a small number of other systems yields best performance. This indicates that the decision tree learner cannot compensate for the noise introduced by many systems in combination. It is possible that one of the systems further down in the F-measure ranking is a good complement for the SMU system, but this is being hidden by the noise. We therefore evaluate the performance of all 21 systems paired with the SMU system; these results are presented in Figure 4.6. In this case, the "frequency" column corresponds to a combination of the SMU and frequency systems, the "CNTS-Antwerp" column corresponds to a combination of the SMU and CNTS-Antwerp systems, and so on. No combination outperforms the SMU-frequency combination, although SMU-BCU and SMU-Sheffield perform better than is indicated by Figure 4.5.

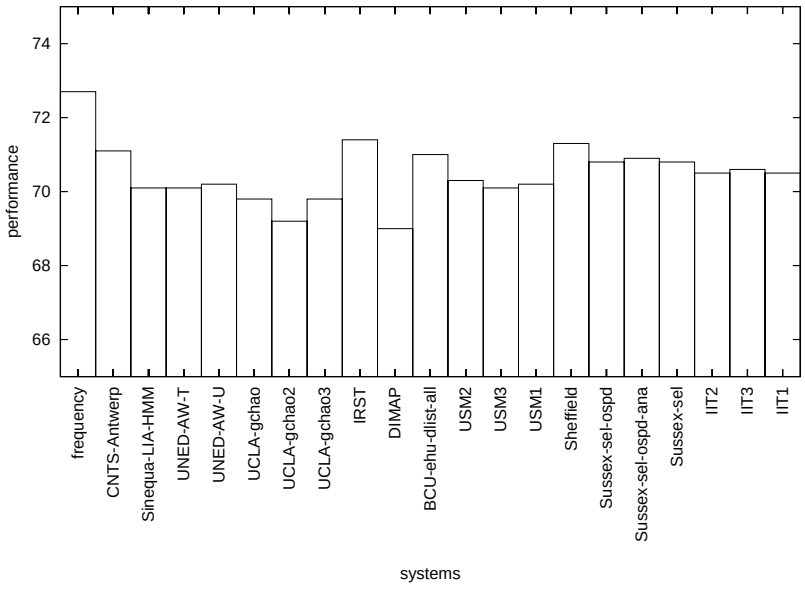

Figure 4.6: Paired System Results

4.5 Summary

We summarize the main findings of this section:

1. Combining a large number of systems leads to too much noise.

2. Combining the first three (F-measure) ranked SENSEVAL systems yields better performance than the current best system (precision 72.69%, recall 72.69%).

We conclude that although decision tree learning seems very well suited to automatically finding complementary systems, this is not necessarily true for WSD when the

combined systems are very intertwined. For future WSD systems, effort may best be spent by creating systems which are independent. As we remarked in section 4.3.2, there are hidden dependencies between features, which makes the task of creating independent systems very hard. For example, the frequency rank feature is not independent of the number of training examples for a word, as a more frequent sense will mean more training examples exist. Thus a supervised system which does not directly use frequency rank will still not be independent from the most frequent sense baseline.

Chapter 5

Combining WSD Knowledge Sources

5.1 Introduction

In this chapter we introduce the notion of a combination WSD system consisting of numerous modules based on different WSD approaches. We motivate each module producing a probability distribution on senses. We present a number of methods for combining WSD knowledge sources, further illustrating the benefits of fully probabilistic systems. Keeping in mind the need to combine such modules, we discuss a number of different approaches to countering data sparseness using smoothing techniques, and find Lidstone's smoothing best suited to our set-up.

5.2 Combination WSD Systems

Stevenson and Wilks (2001) noted that combinations of WSD systems usually outperform their component systems. Florian and Yarowsky (2002) elaborate on the reasons for this increase in performance:

- Using a number of systems may improve robustness. Not all systems attempt all words (see Sections 2.5.2 and 2.5.1), but they may be confident of the ones they do attempt.

- It may be possible to create independent modules for WSD (arising from orthogonal feature spaces). In this case it would be easier to treat each module individually.

- If we combine N WSD systems which have uncorrelated and unbiased errors, the overall annotation error can be reduced by a factor of $\frac{1}{N}$, if these are optimally combined (Perrone and Cooper, 1993), (Dietterich and Kong, 1995).

We implement a combination WSD system – our system consists of a number of self-contained modules each producing a probability distribution on senses (for more details on module implementation see Chapter 6). These modules can output two different probability distributions on senses: one suitable for combining using the Dempster-Shafer method and the other suitable for combination using the method of naive Bayes. For example the part of speech module for the word w produces either $P(s_i|t)$ or $P(t|s_i)$ (see

below) which need to be treated differently during module combination (see Section 5.3) and yield different performances (see Chapter 8).

$$P(s_i|t) = \frac{\text{\# occur. of } w \text{ in sense } s_i \text{ when the PoS tag of } w \text{ is } t}{\text{\# occur. of } w \text{ with PoS tag } t} \quad (5.1)$$

$$P(t|s_i) = \frac{\text{\# occur. of } w \text{ in sense } s_i \text{ when the PoS tag of } w \text{ is } t}{\text{\# occur. of } w \text{ in sense } s_i} \quad (5.2)$$

We will now explain the various methods for combining modules which produce such probability distributions.

5.3 Combining Modules

Combination WSD systems often do not use theoretically motivated methods for combining the different approaches. We begin this section with a brief discussion of related work on combining modules, after which we will present two methods for combining the probabilistic modules introduced in Section 5.2. There are three major ways to combine WSD systems (Xu et al., 1992):

1. Majority voting chooses the sense which is suggested by most systems.

2. Re-ranking is only applicable when systems output a ranked subset of possible senses. The senses in the union of the systems' output are re-ranked according to their rank in the individual systems.

3. Combining posterior sense probability distributions can only be used when the systems produce probability distributions. This is often done using linear interpolation. For details of linear interpolation refer to Section 5.4.

Initial investigations of combination WSD systems relied on voting methods; for example, Kilgarriff and Rosenzweig (2000) use various types of voting to combine answers from SENSEVAL-1 systems to obtain a new system. Majority voting was also chosen by Pedersen (2000) when combining a number of Naive Bayes classifiers, since it has a high empirical performance. However, after initial investigations with majority voting, Stevenson and Wilks (2001) found better performance was attained with memory-based learning in choosing a sense from a combination of selectional preferences, subject codes and collocation feature vectors. Word senses from training data are given to the memory-based learner as vectors, and a sense is assigned to a target word by constructing an unclassified vector and finding the most similar vector from the training data. A more thorough investigation of combination methods is presented in Florian and Yarowsky (2002), who combine six different classifiers. They found that rank-based combination (where each classifier votes for each sense with a weight inversely proportional to the rank assigned) and performance-based voting perform best.

In the following subsections, we present two theoretically motivated methods for combining modules, one for each type of probability distribution introduced in Section 5.2. Our combination methods need to satisfy the following requirements:

1. The method needs to produce a probability distribution (so for example, majority voting is not suitable).

2. The method must use the probability distributions produced by the invoked modules in the combination.

3. Most importantly, the method needs to cope with probability distributions over senses and distributions over subsets of senses (such as produced by a module which assigns probabilities to parts of speech, not individual senses).

These conditions are satisfied by linear interpolation, the Dempster-Shafer theory (Shafer, 1976), and by Bayes Rule yielding three combination methods.

5.3.1 Combining Modules using Linear Interpolation

Our initial approach to combining probability distributions is weighted linear interpolation. This combination method allows us to investigate how a simple approach fares against the more complicated approaches described later in this chapter. Given probability distributions $\mathbf{d_1}, \ldots, \mathbf{d_n}$ generated by n modules, and $\alpha_1, \ldots, \alpha_n$ non-negative weights such that $\sum_{i=1}^{n} \alpha_i = 1$, we form the resulting probability distribution d using the formula

$$\mathbf{d} = \alpha_1 \mathbf{d_1} + \ldots + \alpha_n \mathbf{d_n}$$

As an exhaustive search of all possible weights α_i is not possible (the search space is too large), these are initialized manually and subsequently optimized to maximize the overall F-measure on a development corpus.

5.3.2 Combining Modules using Dempster-Shafer

Next we present the combination method for the $P(s_i|t)$ form of probability distribution over senses, which is based on the Dempster-Shafer theory (Shafer, 1976).[1] In particular, we set the universe U to be the set of senses of the target word, the theory works on subsets of U, satisfying point 3 of the conditions above. We will introduce the theory in the case where only two modules are being combined, present the general form, and explain the simplification which applies in our case. This theory has been frequently used in, for example, user and student modeling (Jameson, 1995), or image segmentation (Bendjebbour et al., 2001).

In common with many combination methods, the Dempster-Shafer theory tacitly assumes that the probabilities being combined are independent – this is an assumption which does not usually hold. However, factoring out dependencies in general is extremely difficult as they are usually hidden. Removing dependencies between the supervised modules would require an enormous amount of training data and storage space. For example, to work out the dependencies between modules based on the preceding and the following word, we would need to store information about the preceding and following word for every target word. This would not be feasible given the amount of training data and the number of modules we are considering.

The foundation of the Dempster-Shafer theory is a probability assignment m on subsets of the universe U. Thus m is a function $m : P(U) \rightarrow [0, 1]$ satisfying:

[1] This work is not the first application of Dempster-Shafer theory to natural language processing: Poznański (1992) applied Dempster-Shafer theory to model the mental states of people making an utterance.

- $m(\emptyset) = 0$
- $\sum_{A \subseteq U} m(A) = 1$

Given m, we can define the *Belief* of a subset A of U as

$$Belief(A) = \sum_{B \subseteq A} m(B)$$

Likewise, the *Plausibility* of the same subset A is

$$Plausibility(A) = 1 - Bel(\bar{A})$$

where $\bar{A}$ is the set theoretic complement of A in U.

Intuitively, the true probability of an event A occurring lies somewhere in the interval

$$[Belief(A), Plausibility(A)]$$

To combine two probability assignments m_1 and m_2, we can use the Dempster Rule of combination (Dempster, 1967):

$$m(A) = \frac{\sum_{\{X,Y|X \cap Y = A\}} m_1(X) m_2(Y)}{1 - \sum_{\{X,Y|X \cap Y = \emptyset\}} m_1(X) m_2(Y)}$$

In general, the Dempster Rule for combining n probability assignments $m_1, \ldots, m_n$ is:

$$m(A) = \frac{\sum_{\{X_1,\ldots,X_n|\cap_i X_i = A\}} \prod_i m_i(X_i)}{1 - \sum_{\{X_1,\ldots,X_n|\cap_i X_i = \emptyset\}} \prod_i m_i(X_i)}$$

We will now present an extended example, which will illustrate the application of the theory for combining our WSD modules. In the WSD domain, an event is a subset of the senses of the target word. Table 5.1 presents the sense information of the word *chew*. Consider the *window* module and assume there is a word in the window which produces the probability distribution in Table 5.2. This gives rise to the function

$$m_1 : P(\{s|s \text{ is a sense of chew}\}) \rightarrow [0, 1]$$

which produces the following probability assignment:

$$\begin{aligned} m_1(\{\text{chew}_{\text{noun}_1}\}) &= \tfrac{1}{6} \\ m_1(\{\text{chew}_{\text{noun}_2}\}) &= \tfrac{1}{3} \\ m_1(\{\text{chew}_{\text{verb}_1}\}) &= \tfrac{1}{2} \\ m_1(x) &= 0 \text{ for any other } x \in P(\{s|s \text{ is a sense of chew}\}) \end{aligned}$$

Note that the function produces the same results as were observed due to the fact that

In this case, since the probability assignment on sets containing multiple senses is zero, *Belief* equals *Plausibility*. For example, the *Belief* and *Plausibility* values for the set $\{\text{chew}_{\text{verb}_1}\}$ (whose set theoretic complement is $\{\text{chew}_{\text{noun}_1}, \text{chew}_{\text{noun}_2}\}$) are calculated as follows:

$$\begin{aligned} Belief(\text{chew}_{\text{verb}_1}) &= \tfrac{1}{2} \\ Plausibility(\text{chew}_{\text{verb}_1}) &= 1 - \left(\tfrac{1}{3} + \tfrac{1}{6}\right) = \tfrac{1}{2} \end{aligned}$$

Sense Id	Gloss
$\text{chew}_{\text{noun}_1}$	biting and grinding food in your mouth
$\text{chew}_{\text{noun}_2}$	a wad of something chewable
$\text{chew}_{\text{verb}_1}$	chew (food)

Table 5.1: Sense information for *chew*

Sense Id	Probability
$\text{chew}_{\text{noun}_1}$	$\frac{1}{6}$
$\text{chew}_{\text{noun}_2}$	$\frac{1}{3}$
$\text{chew}_{\text{verb}_1}$	$\frac{1}{2}$

Table 5.2: *Window* module assignment for *chew*

Suppose there is a sentence which gives rise to the probability assignment in Table 5.3. The corresponding probability assignment is as follows:

$$\begin{aligned} m_2(\{\text{chew}_{\text{noun}_1}, \text{chew}_{\text{noun}_2}\}) &= \tfrac{4}{5} \\ m_2(\{\text{chew}_{\text{verb}_1}\}) &= \tfrac{1}{5} \\ m_2(x) &= 0 \text{ for any other } x \in P(\{s | s \text{ is a sense of chew}\}) \end{aligned}$$

PoS	Prob
Noun	0.8
Verb	0.2

Table 5.3: Part of speech information for *chew*

In this case, the *Belief* and *Plausibility* values for $\text{chew}_{\text{noun}_2}$ are more interesting:

$$\begin{aligned} Belief(\text{chew}_{\text{noun}_2}) &= 0 \\ Plausibility(\text{chew}_{\text{noun}_2}) &= 1 - \left(\tfrac{1}{5}\right) = \tfrac{4}{5} \end{aligned}$$

These are effectively pessimistic and optimistic views of the information provided by the tagger: although we know that the word is being used as a noun with a $\frac{4}{5}$ probability, in one extreme, it may be the case that $\text{chew}_{\text{noun}_1}$ has probability $\frac{4}{5}$ and $\text{chew}_{\text{noun}_2}$ has probability 0, thus the belief of $\text{chew}_{\text{noun}_2}$ is 0 (the pessimistic view). In the other extreme case, the whole of the $\frac{4}{5}$ probability may be referring to the $\text{chew}_{\text{noun}_2}$ sense, with $\text{chew}_{\text{noun}_1}$ having 0 probability (the optimistic view), leading to plausibility of $\frac{4}{5}$.

We use the general form of the Dempster Rule of combination to combine our modules (see Chapter 6 for a presentation of our modules). Given that some of these modules produce probability distributions just on individual senses, the resulting *Belief* will be equal to *Plausibility*, giving us a probability distribution on individual senses as a result of the combination. A possible disadvantage of the $P(s_i|t)$ type of probabilities is they may favour more frequent senses as these appear more often than the less frequent senses, no matter in what context. The following Bayes Rule combination counters such a bias.

5.3.3 Combining Modules using Bayes Rule

Our third combination method for probability distributions, this time for distributions of the form $P(t|s_i)$, is based on Bayes Rule:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

As a consequence of Bayes Rule, the probability distributions for individual modules are multiplied together to yield a final probability distribution on senses, which can be later converted into a single sense assignment or used directly in an application.

Suppose we have n modules and the ith module observes context c_{ik_i} around the target word w. The probability that the word w has the sense s_j can be calculated as:

$$\begin{aligned} P(s_j|c_{1k_1} \cap \ldots \cap c_{nk_n}) &= \frac{P(s_j \cap c_{1k_1} \cap \ldots \cap c_{nk_n})}{P(c_{1k_1} \cap \ldots \cap c_{nk_n})} \\ &= \frac{P(c_{1k_1} \cap \ldots \cap c_{nk_n}|s_j)P(s_j)}{P(c_{1k_1} \cap \ldots \cap c_{nk_n})} && \text{Bayes Rule} \\ &= \frac{\prod_i P(c_{ik_i}|s_j)P(s_j)}{P(c_{1k_1} \cap \ldots \cap c_{nk_n})} && \text{Independence} \end{aligned}$$

This is the general equation. In the case where we combine the *frequency module* with the *basic part of speech module*, the formula will have the form:

$$P(s_j) = P(s_j|\text{pos}(w) = \text{pos}(s_j))P(\text{pos}(s_j) = \text{pos}(w))$$

Note that we again assume independence between modules, however Bayes Rule helps us remove the frequency bias from the supervised modules, as we are considering the probability that a particular context is employed with the given sense, rather than the probability that the given sense is employed with a particular context. Frequency information can be restricted to just one module.

We illustrate our combination method on two modules, based on the preceding and the following word. Consider the word *admirable*, which has two adjectival senses *estimable* and *pleasing* (for definitions see Table 5.4). The probability combination is presented in Table 5.5; the table contains smoothed values for the component probabilities $P(s_i)$ based on the relative frequency of the senses, and $P(l|s_i)$, the probability of lemma l given the sense s_i from the preceding and following word modules. The combination is simply implemented by multiplying the probability distributions together and normalizing the result. The table presents the two fragments *most admirable american* and *most admirable quality*, the *estimable* sense of *admirable* in the first fragment has the probability:

$$\frac{\frac{5}{7} * \frac{2}{3} * \frac{2}{3}}{\frac{5}{7} * \frac{2}{3} * \frac{2}{3} + \frac{5}{7} * \frac{2}{3} * \frac{1}{3} + \frac{2}{7} * \frac{1}{3} * \frac{1}{3} + \frac{2}{7} * \frac{1}{3} * \frac{2}{3}} = \frac{5}{9}$$

5.4 Smoothing

It is necessary to smooth the probability distributions produced by each module – if we only observed a word once in the training corpus, this will give probability one to all

Word	Sense	Definition
admirable	estimable	Deserving of the highest esteem or admiration.
admirable	pleasing	Inspiring admiration or approval.

Table 5.4: Definitions for senses of *admirable*

Sense s_i	$P(s_i)$	Preceding word		Following word		$P(s_i \mid l_1 \cap l_2)$
		l_1	$P(l_1 \mid s_i)$	l_2	$P(l_2 \mid s_i)$	
estimable	5/7	most	2/3	american	2/3	5/9
				quality	1/3	5/18
pleasing	2/7	most	1/3	american	1/3	1/18
				quality	2/3	1/9

Table 5.5: Combination of modules for the word *admirable*

the observed patterns of that occurrence. This would mean that we could never assign a different sense to this word. However, we do not want to discard the one instance. In the case of infrequent words, this may be the only context the word is ever used in. We therefore need to smooth the frequency counts obtained to allow all possible senses, but to still favour the observed patterns taking into account our confidence in the given module.

The chosen method of smoothing needs to satisfy the following requirements:

1. The smoothing value for each module should reflect the confidence we have in that module.

2. It needs to be possible to make smoothing word specific: it is possible that a module is very good at resolving particular words, but not others.

3. The less confident we are in a module, the more the output probability distribution should resemble a uniform distribution over the possible senses, so as not to affect the result (see Section 5.3).

4. The probability distribution generated by trained modules for words not seen in the training corpus should be uniform.

The points above illustrate that we should not smooth with a Zipfian (zeta) distribution; a Zipfian distribution (defined in Section 7.3) would favour the more frequent senses (for which there is an explicit *frequency module*), thus making senses other than the most frequent sense extremely unlikely to be favoured. The method for combining modules which we employ allows us to add in a module which produces a uniform distribution without the new, uninformative module having any effect on the resulting probability distribution on senses. Therefore the requirement for approximating a uniform distribution in points 3 and 4, when the modules do not provide very reliable information, suggests that the best smoothing method is based on a linear interpolation with a uniform prior. It is the linear interpolation weights which allow us to reflect the confidence in different modules, and so the method satisfies point 3.

Smoothing methods based on discounting, which decrease the frequency of data observed in the training phase by a small amount and distribute the discounted frequency between the unseen events (e.g. Good (1953)), would not allow us to approximate a uniform distribution when we are not confident in a particular module.

Treating point 2 above, we specialize to a particular word w, and use Lidstone's smoothing (e.g., Manning and Schütze (1999)):

$$P(s_i|context) = \frac{C(s_i \cap context) + \lambda}{C(w \cap context) + \lambda N}$$

where w is a word with N senses, $s_1, \ldots, s_N$. The C function represents the frequency function and λ is the smoothing value. This is equivalent to a linear interpolation method:

$$P(s_i|context) = \mu \frac{C(s_i \cap context)}{C(w \cap context)} + (1 - \mu)\frac{1}{N}$$

where

$$\mu = \frac{C(w \cap context)}{C(w \cap context) + \lambda N}$$

This smoothing method is frequently criticized for assigning too much probability mass to unseen events. This occurs when N is much greater than $C(w \cap context)$. However, the more senses a word has, the more frequent the word tends to be, thus ameliorating the problem in our application.

To satisfy point 1 above, that each module should be smoothed according to our confidence in it, we want to select smoothing values such that the performance of the system is maximized. As the modules perform with different accuracies on individual words, different smoothing values are found for the most frequent words in the test corpus. We split our training corpus (SEMCOR, and the English lexical sample data) into two parts: $\frac{9}{10}$ for training from which we acquire the frequency counts (such as $C(s_i \cap context)$); $\frac{1}{10}$, the development corpus, for setting the smoothing values (Chen and Goodman, 1996). The optimized value of λ provides an estimate for the confidence in the modules (the higher the value of λ, the lower the confidence in the module). Using linear interpolation as a smoothing method means that the higher the value of λ for a module the more the output probability distribution from that module will resemble a uniform probability distribution.[2]

Figure 5.1 shows the typical shape of a performance against smoothing value graph. As the smoothing value increases, the performance rises sharply reaching a maximum, after which it decreases steadily. However, it is infeasible to exhaustively search all possible smoothing values. We therefore start with a high smoothing value, which we repeatedly halve until performance on the development corpus starts decreasing. The smoothing value (λ) which yielded the highest performance is then used in the testing phase.

The above algorithm gives us a smoothing value for each word occurring in the development corpus. However, we also need smoothing values for words which are not present in this corpus. These are obtained by finding the optimal smoothing value (the one yielding the highest performance) on all ambiguous words which have not been trained

[2]Note that in this section we are dealing with smoothing only, we are not discussing a method for combining modules.

Figure 5.1: Performance against smoothing value

individually. If a word did not appear in the training corpus, its probability distribution becomes the uniform distribution, thus not affecting the choice of sense (see Section 5.3).[3]

5.5 Summary

We have discussed the advantages of combination systems, and outlined our modular probabilistic WSD system. We presented a number of approaches to smoothing and selected Lidstone's smoothing as the most suitable. Three methods for combining modules producing probability distributions over senses were presented.

[3]If a word does not appear in the training corpus at all, all the trained modules will produce a uniform probability distribution and we will be relying on the *frequency* and *basic part of speech* modules to carry out a sense annotation.

Chapter 6

Probabilistic WSD Modules

6.1 Introduction

We present the modules introduced in Chapter 5, which form our probabilistic WSD system. These modules are based on known WSD algorithms, which are modified to be self-contained and to produce a probability distribution on senses. We illustrate the expected functionality of each module with examples taken from our test corpus.

6.2 WSD System

Our probabilistic WSD system is implemented within a modular framework. It is composed of a number of probabilistic WSD components (modules), each producing a probability distribution on senses. The main advantage of this design is its flexibility: it is possible to add or remove modules very easily. A modular setup allows a systematic exploration of combinations of modules to optimize performance.

Our modules are based on parts of known WSD algorithms, which we re-implement. Each module is modified to be self-contained and to produce a probability distribution.[1] We draw from the work of Yarowsky (2000), Mihalcea and Moldovan (2002) and Pedersen (2002), and we also implement a number of baseline modules.

6.3 Unsupervised Modules

We use two types of modules in our WSD system: unsupervised and supervised. In this work, we define an unsupervised module to be a module which our system does not need to train (in particular, we classify the module based on WordNet frequency as unsupervised).

[1] A probability distribution is created over WordNet senses, as well as an extra sense 'none', not present in WordNet. Without the extra sense, each word present in WordNet would have to be assigned a WordNet sense. However, WordNet only covers nouns, verbs, adjectives and adverbs. So a word like *no* could never take the 'negation' sense (which does not appear in WordNet) rather than the less frequent 'number' sense (which is listed in WordNet). For the negation *no*, the probability of the 'none' sense is close to 1 and the noun sense's probability is very small resulting in the program selecting the 'none' sense.

6.3.1 *Frequency* Module

We have seen in the description of the SENSEVAL-2 English all words task (Section 2.5.2), that it is quite difficult to beat the most frequent sense baseline (a system selecting for each word its most frequent sense). We reason that the most frequent sense provides a good default value for words which do not obviously (due to another module) have another sense, and thus a most frequent sense module forms a part of our system. As a side effect, with this module in place, it is trivial to evaluate the performance of the baseline on each of the data sets.

Usually a most frequent sense baseline returns the most frequent sense (derived from some other corpus) given the word's (known or guessed) part of speech. So if we know that in our text the word *dog* is always used as a noun, this module will always return the domestic dog sense of the word (rather than another noun sense e.g., the unpleasant woman sense).

We require all of our modules to produce a probability distribution. WordNet, the dictionary we employ in this work, contains two types of information we could use for this purpose: a manual ranking of senses created by the WordNet lexicographers on the basis of their perceived frequency in English, and the frequency of occurrence of each sense within the genre balanced SEMCOR corpus. Each information type can be converted into a probability distribution.

The manual ranking of senses could be used in conjunction with Zipf's law to yield a distribution on senses (given the part of speech). Zipf's law (Zipf, 1949) provides the following probability distribution over ranked senses

$$f_r = ar^{-b}$$

where f_r refers to the frequency of the rth ranked sense and a, b are such that

$$\sum_{i \in \{\text{senses}\}} f_{\{r:r \text{ rank of sense } i\}} = 1$$

The shape of a Zipfian distribution can be seen in Figure 7.1. However, some words are exceptions to Zipf's law; the SEMCOR frequency of their topmost senses is quite close (e.g., *bear* which has 15 senses, and the top three verb senses have frequencies 25, 17, and 13). In this case, the probability distribution arising from Zipf's frequencies would be very different to the observed one, and so this method is insufficient.

Making use purely of the SEMCOR frequency counts is also not sufficient. A probability distribution in this case would be obtained by normalising the SEMCOR frequencies within each PoS. However, it is possible for multiple senses to have identical SEMCOR frequency (often zero). The frequency module would then not distinguish between such senses.

We would therefore like to use the manual ranking as a tie-breaker for the frequency counts. We achieve this by smoothing the frequency counts of the word w with n senses $s_1, \ldots s_n$ by the inverted rank (ir) as follows:

$$f(s_i) = o(s_i) + ir(s_i)$$

where $o(s_i)$ is the observed SEMCOR frequency count and $ir(s_i) = n + 1 - r(s_i)$, with $r(s_i)$ being the rank of the sense s_i. This is illustrated in Table 6.1: the word *abbey*

appeared once in its most frequent sense and neither of its other senses were seen. After smoothing, no two frequencies are the same.

Additionally, in the case of the *frequency module*, it only makes sense to consider the probability of a sense s_i given the part of speech of this instance of w is $\text{pos}(s_i)$. Thus, the probability of the sense s_i of the word w is given by:

$$P(s_i|\text{pos}(s_i)) = \frac{f(s_i)}{\sum_{\{j:\text{pos}(s_j)=\text{pos}(s_i)\}} f(s_j)}$$

where $f(s_i)$ is the smoothed SEMCOR frequency of the sense s_i. In our example, this corresponds to the sense frequencies being divided by $4 + 2 + 1 = 7$.

Sense	Rank	Frequency
Original Entry:		
abbey%1:06:02::	1	1
abbey%1:06:01::	2	0
abbey%1:06:00::	3	0
Modified Entry:		
abbey%1:06:02::	1	1 + (3 + 1 - 1) = 4
abbey%1:06:01::	2	0 + (3 + 1 - 2) = 2
abbey%1:06:00::	3	0 + (3 + 1 - 3) = 1

Table 6.1: WordNet distribution information

6.3.2 *Basic Part of Speech* Module

WSD systems often make use of the part of speech (PoS) information of words; even the *frequency* module requires information about each word's PoS to work effectively. This is usually obtained from a tagger, which is run as a pre-processing stage of a WSD system. Often, the tagger is used as a filter: all senses not compatible with the chosen PoS are entirely removed from consideration. However, this means that if a word is assigned a wrong PoS, there is no scope for recovering from the error. We instead use the HMM tagger due to Elworthy (1994), which produces a posterior probability distribution on CLAWS-II PoS tags using the Forward-Backward algorithm.

There are 166 tags in the CLAWS-II tagset; it contains tags such as NN1 (singular commons nouns), NP (noun proper, neutral for number), or VB0 (to, base form), VVZ (-s form of a lexical verb). For each word w, the tagger returns a probability distribution on these tags. However, WordNet senses are only divided up into four categories: noun, verb, adjective, and adverb. We therefore sum the probabilities obtained over all the noun PoS tags to obtain a probability of the word w being used as a noun, the probabilities of the verb PoS tags make up the probability of w being used as a verb, and so on. The CLAWS-II tagset also contains tags which do not describe nouns, verbs, adjectives or adverbs, such as II (preposition). The probabilities of these 'other' tags make up the probability of the word w having a non-WordNet PoS (and so being used in the extra 'none' sense). The full 166 tag tagset is used in some of the subsequent modules (Sections 6.4.1 and 6.4.3).

We illustrate the module on the sentence *She danced with abandon*, where the PoS tagger assigns a high probability to the NN1 tag of the word *abandon*. The original PoS distribution obtained from the tagger can be seen in Table 6.2. The simplified PoS probabilities over the WordNet (noun, verb, adjective, adverb and other) classes are shown in Table 6.3. In this table, the verb tag probabilities of the word *danced* (VVD and VVN) have been merged into a verb probability. Smoothing (Section 5.4) is applied to all PoS categories containing at least one sense of the word (for example, to the noun and verb categories for the word *danced*), as well as to the 'other' category (to allow for senses not in WordNet).

Word	Tag	Prob	Tag	Prob
She	PPHS1	1	–	–
danced	VVD	0.98	VVN	0.02
with	IW	0.95	II	0.05
abandon	NN1	0.96	VV0	0.04

Table 6.2: Original PoS distribution

Word	Noun	Verb	Adjective	Adverb	Other
She	–	–	–	–	–
danced	0.00	1.00	0.00	0.00	0.00
with	–	–	–	–	–
abandon	0.96	0.04	0.00	0.00	0.00

Table 6.3: Simplified PoS distribution

Note that in this module, the probability distribution generated is over parts of speech. If we had instead generated the distribution over senses, this would have had the effect of penalizing senses in the more common parts of speech for the word. For example, consider a word with four noun senses and one verb sense. If the noun part of speech is assigned a probability of 3/4 by the tagger and the verb PoS is given 1/4, our system will assign the probability of 3/4 to all the noun senses, rather than producing a probability distribution on senses and thus assigning each noun sense 3/16 ($< 1/4$).

6.4 Supervised Modules

The *frequency* and the *basic part of speech* modules do not need to be trained (at least, not by the WSD system). The remaining modules all require training data and produce probability distributions on senses of one of the following two types:

$P(\text{sense} \mid \text{observed training data})$ (Dempster-Shafer combination)

$P(\text{observed training data} \mid \text{sense})$ (Bayes Rule combination)

The forms of the distribution produced are dictated by the method which is used to combine modules together (Section 5.3). We now present the trained modules, many of which are based on known successful WSD approaches.

6.4.1 *PoS Context* Modules

The contextual *part of speech* modules work on the principle that PoS tags of words in the context of the target word are informative. Attention needs to be paid to the size of the tagset for this module: if it is too small, it will not be very informative, if it is too large, it will require too many examples to train. We use the 166 tag CLAWS-II tagset to create five *part of speech* modules: part of speech tag of the words one and two to the left (*pos-1*, *pos-2*) and right (*pos1*, *pos2*) of the target word and the word's own tag (*pos0*). In the case of the *pos0* module, we generate the following two types of probability distributions:

1. The probability of the sense being s_i given that the tag of the word w is t is:

$$P(s_i|t) = \frac{\text{no. of occurrences of } w \text{ in sense } s_i \text{ when the PoS tag of } w \text{ is } t}{\text{no. of occurrences of } w \text{ with PoS tag } t}$$

2. The probability of the tag being t given that the sense of the word w is s_i is:

$$P(t|s_i) = \frac{\text{no. of occurrences of } w \text{ in sense } s_i \text{ when the PoS tag of } w \text{ is } t}{\text{no. of occurrences of } w \text{ in sense } s_i}$$

For the *pos1* module we would consider the tag of the word immediately following w. We present three frequency distributions from the *pos0* module for the word *shirt* ($P(\text{shirt}_i|\text{pos0} = \text{NN1})$, $P(\text{shirt}_i|\text{pos0} = \text{NN2})$, and $P(\text{shirt}_i|\text{pos0} = \text{VVD})$, and the corresponding Bayes Rule probabilities) in Table 6.4. In this table, $f(s \cap t)$ denotes the number of occurrences of w in the given sense with the given PoS tag t, and $f(s)$ is the number of occurrences of *shirt* in the given sense.

Sense (s_i)	PoS (t)	$f(s \cap t)$	$f(s)$	$f(t)$	$P(s_i\|t)$	$P(t\|s_i)$
shirt - noun	NN1	8	9	8	$\frac{8}{8}$	$\frac{8}{9}$
shirt - verb	NN1	0	1	8	0	0
shirt - noun	NN2	1	9	1	$\frac{1}{1}$	$\frac{1}{9}$
shirt - verb	NN2	0	1	1	0	0
shirt - noun	VVD	0	9	1	0	0
shirt - verb	VVD	1	1	1	$\frac{1}{1}$	1

Table 6.4: Part of speech distributions for the word *shirt*

6.4.2 *Window* Module

Words in the context of the target word have been shown to have disambiguating power (Gale et al. (1992a)). For example, if the noun *bank* is seen in the context of the word *money*, it is much more likely to have its financial institution noun sense. It is therefore

informative to look for certain sense indicative words occurring in a window around the target word. Gale et al. use a ±50 word context in their disambiguation system.

In the training phase of this module, frequencies of words up to 50 positions to the left and right of each target word (100 words in total) are stored along with the sense of the target word. After removing stoplist words, each of the context words is treated as 'indicating' the given sense of the target word.[2] For each target word w, the training phase results in a number of probability distributions given each sense s_i arising from each of the context words c, the two pairs of probability distributions being as follows:

1. The probability of the sense being s_i given that the context of the word w is c is:

$$P(s_i|c) = \frac{\text{no. of occurrences of } c \text{ in the context of } w \text{ in sense } s_i}{\text{no. of occurrences of } c \text{ in the context of } w}$$

2. The probability of the context being c given that the sense of the word w is s_i is:

$$P(c|s_i) = \frac{\text{no. of occurrences of } c \text{ in the context of } w \text{ in sense } s_i}{\text{no. of occurrences of } w \text{ in sense } s_i}$$

6.4.3 *PoS Trigram* Module

The *trigram* module is very similar to the *window* module described in the preceding section. It is also based on co-occurrence information, this time looking at a group of three adjacent words. A number of studies (e.g., Choueka and Lusignan (1985)) have shown that humans only require a very small context of two or three words for disambiguation, justifying the small window size. Trigrams have been successfully employed for WSD by Pedersen (2002). The main defect of the method is its sparse data problem, which is even worse than in the *window* module as we are now concerned with co-occurrences of three words, not just the presence of one word in the vicinity of another.

Using the N-gram Software Package created by Pedersen,[3] we implemented a *trigram* module based on part of speech tags. We use PoS tags to abstract away from the more usual word form trigrams, which suffer from sparse data problems. In this module, each sense tagged word w from the training corpus is a part of three trigrams:

Position -1: w is the first word of the trigram.

Position 0: w is the middle word.

Position 1: w is the last word of the trigram.

Table 6.5 contains an example for the word *Algerian* in the sentence (the relevant PoS tags follow an underscore):

... to_II the_AT Algerian_JJ rebels_NN2 entails_VVZ ...

We produce the following probabilities:

[2]Note that removing stoplist words is only done for efficiency reasons – since they do not usually indicate any particular sense, they would be found to be uninformative in disambiguation. The case where a neighbouring stoplist word contributes information (such as in the case of *believe in*) will be dealt with by the *lemma* module introduced in Section 6.4.4. There is no frequency cutoff for the *window* module, all non-stoplist words are included.

[3]NSP is available from `http://www.d.umn.edu/~tpederse/code.html`

Sense (s_i)	Trigram (t)			$f(s_i \cap t)$	$f(s_i)$	$f(t)$	$P(s_i\|t)$	$P(t\|s_i)$
Position: -1								
algerian - adjective	JJ	NN2	VVZ	1	3	1	1	$\frac{1}{3}$
algerian - noun	JJ	NN2	VVZ	0	0	1	0	0
Position: 0								
algerian - adjective	AT	JJ	NN2	1	3	1	1	$\frac{1}{3}$
algerian - noun	AT	JJ	NN2	0	0	1	0	0
Position: 1								
algerian - adjective	II	AT	JJ	1	3	1	1	$\frac{1}{3}$
algerian - noun	II	AT	JJ	0	0	1	0	0

Table 6.5: Frequency trigram data for *Algerian*

1. The probability of a sense s_i given the trigram t involving the word w is given by:

$$P(s_i|t) = \frac{\text{no. of occurrences of } w \text{ in sense } s_i \text{ in trigram } t}{\text{no. of occurrences of } w \text{ in trigram } t}$$

2. The probability of a trigram t given the sense s_i of the word w is given by:

$$P(t|s_i) = \frac{\text{no. of occurrences of } w \text{ in sense } s_i \text{ in trigram } t}{\text{no. of occurrences of } w \text{ in sense } s_i}$$

In our example, all three trigrams, positions -1, 0, and 1, give rise to higher probabilities for the adjectival sense (indicated by %3 in its senseid) of the word *Algerian*.

6.4.4 *Lemma Co-occurrence* Module

Although we do not have enough data to train trigrams on surrounding lemmas, bigrams (two adjacent words) have also been found useful in WSD. Bigrams represent a large part of the local context which was found sufficient by human annotators (Choueka and Lusignan, 1985), and in certain cases may be more informative than the *window* module. For example, the fact that the word *behind* occurs within the ±50 word window of the target word is uninformative, whereas if we know that it immediately precedes the word *bars*, the sense of *bars* is clear.[4]

We therefore create a co-occurrence module by extracting lemmas surrounding our target word (we focus on words one and two to the left, *lemma-1*, *lemma-2*, *lemma-3* and right, *lemma1*, *lemma2*, *lemma3* of the target word). We use the frequencies of co-occurrence to produce a probability distribution; for *lemma-1* the two probabilities are:

1. The probability of the target word w_t being preceded by the lemma l_p given that w_t has sense s_i:

$$P(s_i|l_p) = \frac{\text{no. of occurrences of } w_t \text{ in sense } s_i \text{ preceded by } l_p}{\text{no. of occurrences of } w_t \text{ preceded by } l_p}$$

[4]Note that in the case of the *lemma* modules, the stoplist is not applied.

2. The probability of the target word w_t being preceded by the lemma l_p given that w_t has sense s_i:

$$P(l_p|s_i) = \frac{\text{no. of occurrences of } w_t \text{ in sense } s_i \text{ preceded by } l_p}{\text{no. of occurrences of } w_t \text{ in sense } s_i}$$

l_p	Sense id (s_i)	$f(s_i \cap l_p)$	$f(s_i)$	$f(l_p)$	$P(s_i\|l_p)$	$P(l_p\|s_i)$
behind	bars%1:06:00::	8	8	8	1	1
behind	bars%1:06:04::	0	2	2	0	0
cocktail	bars%1:06:00::	0	8	8	0	0
cocktail	bars%1:06:04::	2	2	2	1	1

Table 6.6: Lemma co-occurrence data for *bars*

An example for the word *bars* and its preceding word can be found in Table 6.6. We can see that if the preceding word is *behind*, the word *bars* is more likely to have the obstructing metal sense (1:06:00::), whereas if the preceding word is *cocktail* the establishment where alcoholic drinks are sold (1:06:04::) is more likely.

6.4.5 *Head* Module

Information can also be gained from the distance to the nearest phrasal head and its type (we only consider noun and verb phrases in this work). We use the Briscoe and Carroll (2002) RASP parser[5] to obtain grammatical relations (GRs) between words in each sentence. The GRs produced include subject, object, modifier, etc. See Figure 6.1 for an example of the sentence

The school grounds are large.

The head of the noun phrase "*The school grounds*" can be found to be *grounds* because it does not modify another word in an *ncmod* relation (unlike the word *school*), and is also modified by a determiner (the *detmod*) relation.

```
(ncsubj are grounds _)
(xcomp _ are large)
(ncmod _ grounds school)
(detmod _ grounds The)
```

Figure 6.1: GRs for *The school grounds are large*

The usefulness of the *head* module is easiest to observe if the following word is the head of a phrase. An example for the word *Anglican* followed by a noun phrase head is shown in Table 6.7. The probabilities generated are:

[5]The RASP parser is available from `http://www.cogs.susx.ac.uk/lab/nlp/rasp/`

1. The probability that word w has sense s_i given that it is followed by a head of an NP is:

$$P(s_i|\text{NP}) = \frac{\text{no. of occurrences of } w \text{ in sense } s_i \text{ followed by an NP head}}{\text{no. of occurrences of } w \text{ followed by an NP head}}$$

2. The probability that word w has sense s_i given that it is followed by a head of an NP is:

$$P(\text{NP}|s_i) = \frac{\text{no. of occurrences of } w \text{ in sense } s_i \text{ followed by an NP head}}{\text{no. of occurrences of } w \text{ in sense } s_i}$$

Sense id (s_i)	$f(s_i \cap \text{NP})$	$f(s_i)$	$f(\text{NP})$	$P(s_i\|\text{NP})$	$P(\text{NP}\|s_i)$
anglican%3:01:00::	0	0	0	0	0
anglican%1:18:00::	6	6	6	1	1

Table 6.7: Head data for *Anglican* + NP head

We can see that the adjective sense (sense id containing %3) is more likely than the noun sense; this is intuitively plausible since the head of a noun phrase must be a noun, and we are more likely to have an adjective preceding a noun. This module trains probabilities based on the word one and two to the left and right of the target word being a head of a noun or a verb phrase.

6.4.6 *Grammatical Relation* Module

Our last module exploits noun–verb relations obtained from the RASP parser, along the lines of Hindle (1990) and Resnik (1992). In the training phase, the GRs of each noun target word are examined and information about it being a subject, direct or indirect object are stored together with the relevant verb. An example is shown in Table 6.8, for the word *attitude*. The table shows the definition of two of the senses of the word along with the WordNet example sentences. The last column shows the information which we would acquire from the sentence.

Sense	Definition	Example sentence	GR
1:04:00	a theatrical pose created for effect	the actor struck just the right attitude	struck dobj
1:09:00	a complex mental state involving beliefs and feelings and values and dispositions to act in certain ways	he had the attitude that work was fun	have dobj

Table 6.8: Grammatical relation for *attitude*

This module is smoothed, but unlike Resnik's work, we do not group together 'similar' verbs. Resnik observed that in its basic form, this approach suffered from the sparse data

problem. This is to be expected as the amount of sense tagged training data is limited. We are further reducing the training data to the sentences that our parser can parse, and then focus only on the words filling the grammatical roles we are interested in. He therefore grouped together similar verbs (verbs in the same WordNet class) and used the training data for each verb in a group as training data for the whole group. Since this is not the only module used in our approach, we do not carry out this generalization and the module returns a uniform distribution on senses when there is no training data for a particular word.[6]

6.5 Summary

Module type	Number of modules
Frequency module	1
Basic part of speech module	1
Part of speech context module	7
Window module	1
Part of speech trigram module	3
Lemma co-occurrence module	7
Head module	5
Grammatical relation module	1

Table 6.9: Number of modules produced

We have described the individual modules in our WSD system, motivating and illustrating each approach where possible. See Table 6.9 for the number of modules produced by each of the methods described in the preceding sections. We explained the modifications we have made to each approach to result in probability distributions. We have linked each module to the two methods of combination presented in Chapter 5, which give rise to two sets of results presented in Chapter 8.

[6]Note that due to sparse data, this module is used very rarely.

Chapter 7

Subcategorization Frame Acquisition

7.1 Introduction

We introduce the notion of a task-based evaluation for WSD systems. Throughout the chapter, we discuss the use of subcategorization acquisition as an evaluation task for WSD systems. We describe the task of subcategorization acquisition, and investigate the sensitivity of Korhonen's (2002) subcategorization acquisition system to word senses, by evaluating the acquired subcategorization frames when more than one sense of a verb is considered during acquisition. We present experiments indicating the suitability of subcategorization frame acquisition as an evaluation method for WSD.

7.2 WSD Evaluation

There are two approaches to evaluating WSD systems: the gold standard evaluation, and the task-based evaluation. Before presenting subcategorization acquisition as a possible evaluation method for WSD, we describe both these evaluation methods.

7.2.1 Gold Standard Evaluation

The gold standard evaluation is the most frequently used evaluation method for WSD systems. A selection of words in a chosen text is manually annotated by lexicographers. An unlabeled version of this text is also annotated by an automatic WSD system, and the resulting annotation is compared to the manually created annotation. Such a comparison results in precision and recall values as defined in Section 2.5.

As long as WSD systems are evaluated on the same annotated corpus, they can be directly compared using the F-measure, which is a weighted combination of precision and recall:[1]

$$\text{F-measure} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

The main disadvantage of gold standard evaluations is their failing to give credit for selecting an incorrect but closely related sense. The SENSEVAL-2 scoring software

[1]This definition of F-measure is equivalent to F_1, a specialization of the F_α measure.

attempted to address this issue: WSD systems were allowed to return a probability distribution on senses, which contribute to the score of the system in proportion to the probability assigned. However, this does not completely solve the problem of words with closely related senses (because to get any credit systems still have to return the correct sense with some probability), and in any case not all systems are capable of returning a probability distribution on senses.

7.2.2 Task-Based Evaluations

To overcome the problem of systems selecting a related, but incorrect sense, Resnik and Yarowsky (1999) suggested employing less fine-grained sense inventories. For example, the verb *get* has 37 senses in the very fine-grained WordNet 1.6, whereas only 23 senses are listed in CIDE+ (Procter, 1995). As we have pointed out in Section 3.2.1, the decision whether to split a sense or not usually rests with the lexicographer – should annotating *get* with sense s_i instead of s_j, when both s_i and s_j are encompassed under sense s in CIDE+, count as a serious mistake?

Resnik and Yarowsky suggest restricting the inventory to only those distinctions which are typically realized cross-linguistically. This is partly implemented in evaluations based on machine translation. In these evaluations, the senses of a word from a source language are its target language translations.[2] A task-based environment provides the ultimate demonstration of success of a WSD technique and allows evaluation of senses that matter for the application in question.

For example, SENSEVAL-2 included one task-based task: the Japanese translation task (Kurohashi, 2002). In this task, an inventory was created for 40 words (20 nouns and 20 verbs), by examining the 100 most frequent phrase unigrams, bigrams and trigrams of each word in a nine year section of the Mainichi Newspaper corpus. These frequent phrases were translated into English by a professional translation company, resulting in a number of translation memory records (pairs of equivalent Japanese and English expressions). The sense inventory consists of the translation memory records, which were used to annotate the test corpus (with associated correctness measures where appropriate).

In a machine translation based evaluation, the performance of the WSD system can be directly measured in the translated text – if the system sense resolved a word wrong with respect to its inventory, the word will be assigned a wrong translation. Since WSD is an intermediate task (Wilks and Stevenson, 1996), direct evaluation of the application which employs a WSD system may be more informative than a gold-standard evaluation. A gold-standard evaluation also does not reveal the effect that changing the inventory will have on the performance of the WSD system, which will happen when the WSD system is eventually employed within an application.

7.3 Subcategorization Frame Acquisition

Subcategorization frames describe the types of complement a verb can take. For example, the following sentences indicate that the word *believe* can select for a noun phrase

[2]The senses which Resnik and Yarowsky will include in their inventory may be even more coarse-grained, since it is possible that a certain language pair preserves fine-grained distinctions but this distinction is not preserved in general.

complement (sentence 1), and a sentential complement (sentence 2).

1. I *believe* it.

2. I *believe* that he will leave.

These SCF frames together with their frequencies are collected in a lexicon, which is used by other natural language processing tools. For example, the SCF lexicon can improve the accuracy of probabilistic parsers (Carroll et al., 1998b).

In this work, we will integrate our WSD with an existing subcategorization frame (SCF) acquisition system. However, we will first describe the need for automatic SCF acquisition and then move onto describing Korhonen's work on SCF acquisition in more detail.[3]

Manual development of large subcategorization lexicons is difficult as subcategorization is not static – subcategorization can change between domains and over time. Manually developed SCF lexicons also do not provide the relative frequencies of different SCFs which are used in probabilistic parsing.

Over the past years, several approaches have been proposed for automatic acquisition of subcategorization from corpus data (e.g., Briscoe and Carroll (1997), Carroll et al. (1998a), Sarkar and Zeman (2000)), but they have been found to perform similarly. Automatic SCF acquisition consists of two phases:

1. Possible SCFs are suggested (hypothesised).

2. Reliable hypotheses are selected.

For all the approaches used, it is the second step which is a significant source of error. The first step suggests SCFs based on potentially noisy data, these SCFs are then smoothed after which the second step removes unreliable SCFs. This is illustrated in Table 7.1: there are m SCF frames in total for a given verb, and only n of them are observed. The frame i has an associated observed probability $op(SCF_i)$ and backoff probability $bp(SCF_i)$. The final probability for the frame i is obtained by adding the observed and backoff probabilities together, and if the resulting $p(SCF_i)$ satisfies some conditions, the SCF i is selected.

We focus on Korhonen's work (Korhonen, 2002) which deals with the smoothing of acquired SCFs. She observed that if smoothing was done by backing-off to the SCF distribution of related verbs rather than using a uniform distribution, the SCF acquisition performance increased even if a simple threshold method was used to select reliable hypotheses.

Korhonen's work is motivated by past work (e.g., Levin (1993)) showing that verbs fall into a number of classes which are distinctive in their subcategorization. She suggests that the backoff could be acquired from SCFs belonging to the target verb's class and these could then be used to smooth the SCF distribution of the target verb. However, the associations have only been shown to exist between SCFs and particular verb senses. This poses a problem for most SCF acquisition systems as they acquire SCFs for words, not for

[3]The work on integrating a WSD system into the SCF system was carried out jointly with Anna Korhonen.

$op(SCF_1)$	+	$bp(SCF_1)$	=	$p(SCF_1)$
$op(SCF_2)$	+	$bp(SCF_2)$	=	$p(SCF_2)$
$op(SCF_3)$	+	$bp(SCF_3)$	=	$p(SCF_3)$
...		...		...
$op(SCF_n)$	+	$bp(SCF_n)$	=	$p(SCF_n)$
0	+	$bp(SCF_{n+1})$	=	$p(SCF_{n+1})$
...		...		...
0	+	$bp(SCF_m)$	=	$p(SCF_m)$

Table 7.1: Smoothing subcategorization frames

senses. Korhonen's system assumes the WordNet (Miller et al., 1990) most frequent sense for each verb,[4] yielding a back-off based on the behaviour of verbs with senses related to the most frequent sense of the target verb. Where the sense is assigned correctly, Korhonen reports significant improvement in acquisition performance: on a set of 45 test verbs, the F-measure (Section 7.5) improves by 17% against the baseline method, which does not assume a sense.

7.4 WSD for SCF Acquisition

Up to this point, we have described Korhonen's system. Starting from this section we will describe our joint work on integrating WSD with SCF acquisition.

Korhonen observed an improvement in acquisition performance, indicating that for each verb there is some single sense which accounts for most of the verb's subcategorization behaviour. This is consistent with the distributions of sense frequencies being Zipfian, which tells us that the ith most frequent sense (i.e., the sense with rank frequency i) will appear in corpora with $1/i^{\alpha}$ frequency of the most frequent sense (for some value of α). The shape of a Zipfian distribution can be seen in Figure 7.1, note the tail approaching zero indicating that many senses of a word are very rare.

A major extension of the present work would be to carry out an experiment to test the statistical hypothesis that the sense frequencies of words in SEMCOR follow a Zipfian distribution. This could be tested by empirically explicitly computing the corresponding value of α. The maximum likelihood would determine whether the hypothesis was accepted, and α would be the 'Zipf value' for sense frequencies in SEMCOR.

However, preliminary experiments have uncovered evidence against this hypothesis for many highly polysemous verbs, for which the distribution of the frequent senses in balanced corpus data tends to be closer to uniform rather than Zipfian (Preiss et al., 2002). In SEMCOR, the frequency of the top ranked senses did not decay as quickly as predicted by Zipf's Law, which indicated the importance of senses other than the most frequent. In fact, the frequency of the top two or three senses of these medium and high frequency verbs was almost equal.[5]

[4]The WordNet most frequent sense is determined by the frequency data in the associated SEMCOR corpus.

[5]Note that this was only found to be true for verbs, whose behaviour was observed to be different to other parts of speech.

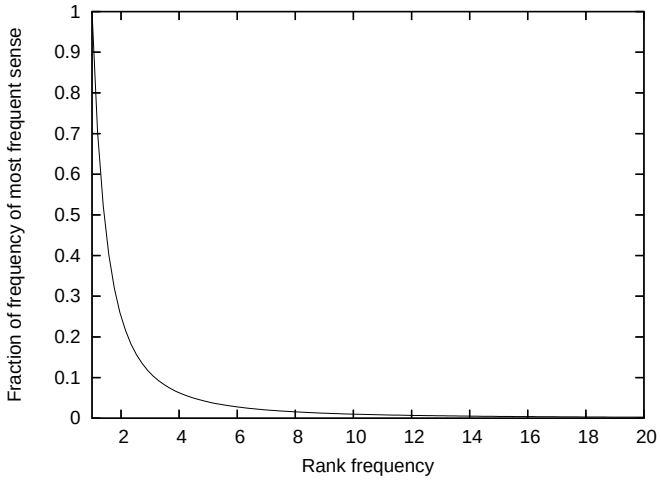

Figure 7.1: Zipfian distribution

Therefore, we concluded that for the important group of medium and high frequency verbs, backing-off to the most frequent sense may not be sufficient in SCF acquisition. To improve the acquisition performance, we suggested considering senses other than merely the most frequent sense.

Korhonen originally found a performance increase "when the most frequent sense was assigned correctly" on her 45 verb corpus, but although the first sense in WordNet was the most frequent in SEMCOR, it may not be the most frequent sense in a different corpus.[6] The frequencies of subcategorization frames have been shown to vary for example across genre, and this variation has been attributed to the change in the sense distribution within different genres (Roland et al., 2000; Roland and Jurafsky, 2001). The ultimate goal of automatic acquisition is to be able to produce domain-specific lexicons tailored to particular applications, which means that the most frequent sense also needs to be detected automatically.

Our aim was to show that automatic WSD improves SCF acquisition in the cases where a verb does not have a clear most frequent sense. Since the system will also detect the most frequent sense for all verbs in the corpus, the performance of the SCF acquisition system is expected to go up in restricted domains where the detected most frequent sense may not match that in WordNet.

To investigate the accuracy of SCF acquisition when multiple senses were considered, we carried out a pilot experiment with manually sense-annotated data (i.e., 100% accurate WSD drawn from SEMCOR) (Preiss and Korhonen, 2002). We now present the baseline system, and explain how information from the WSD system is incorporated in the SCF acquisition system. Experimental results on 29 difficult verbs (i.e., verbs where the top ranked senses tend to occur equally frequently in a balanced corpus) are presented and discussed in Section 7.5.

[6]For example, consider the British English and American English senses of the noun *line*. The most frequent sense of the word in American English is the queue sense, which does not occur much in British English.

7.4.1 Baseline System

Levin (1993) has demonstrated that verb *senses* divide into semantic classes, which are distinctive in terms of subcategorization. Korhonen (2002) shows that many verb *forms* (actual words) also divide into such classes, according to their most frequent sense. For instance, the verb form specific SCF distributions for *fly* and *move* correlate quite closely because the most frequent senses of these verbs (according to the WordNet frequency data) are similar. They both belong to Levin's "Motion verbs".

Korhonen's extension of the Briscoe and Carroll's (1997) SCF acquisition system works by first identifying the sense (the semantic class) for a predicate. The semantic class for a verb is found by extracting the most frequent sense for the given verb from WordNet. This WordNet sense is mapped onto a Levin class.[7] The resulting semantic classes are based on Levin classes (Levin, 1993); mostly on broad classes (e.g., 51. "Motion verbs") rather than subclasses (e.g., 51.2 "*Leave* verbs").[8]

After the semantic class is identified, the system of Briscoe and Carroll (1997) is used to hypothesise a SCF distribution from corpus data. This system employs a robust statistical parser (Briscoe and Carroll, 2002) and a comprehensive classifier which is capable of distinguishing 163 verbal SCFs – a superset of those found in the ANLT (Boguraev and Briscoe, 1987) and COMLEX Syntax dictionaries (Grishman et al., 1994).

The SCF distribution is smoothed using the back-off estimates of the semantic class of the verb. The smoothing method is linear interpolation (e.g., see the work of Chen and Goodman (1996)). The back-off estimates are obtained using the following method:

1. 4–5 representative verbs are chosen from a verb class.[9] These verbs were chosen subject to the constraint that enough data was found for each verb (around 300 occurrences).

2. SCF distributions are built for these verbs by manually analysing around 300 occurrences of each verb in the British National Corpus (BNC) (Leech, 1992). So for each verb, a simple listing of SCFs and their frequencies will be produced.

3. The resulting SCF distributions are merged (to produce a generic back-off for the class) using simple linear interpolation, giving equal weight to each distribution.

For example, the back-off estimates for the "Motion verb" *fly* are constructed by merging the SCF distributions for five other "Motion verbs": *move*, *slide*, *arrive*, *travel*, and *sail*.

As a final step, a simple empirically-determined threshold is used on the smoothed estimates for each frame (yielded from their merged frequency counts) to filter out SCFs which the statistical parser had incorrectly associated with particular verbs.[10]

[7] See the work of Korhonen (2002) for details of the mapping.

[8] The broad classes are more useful because they allow adequate generalizations to be made, but are still distinctive enough in terms of subcategorization to provide good accuracy.

[9] The verb for which subcategorization is being acquired is always excluded.

[10] This filtering was found to outperform binomial filtering, see Korhonen (2002) for a detailed discussion.

7.4.2 Combining with WSD

Our initial experiments with 29 difficult verbs (verbs whose top senses are close in frequency in SEMCOR) with manually sense annotated text involved modifying Korhonen's baseline system described in Section 7.4.1 so that it could benefit from sense information for the verbs it was acquiring SCFs for. The motivation for this work is that a verb's SCF distribution is made up of the SCF distributions for each sense.

We created different corpus datasets for each sense being disambiguated, grouped any remaining senses together and created a communal dataset for these.[11] SCFs were acquired separately from these datasets, and these were smoothed with back-off estimates of the relevant sense. No smoothing was done in the case of the dataset of grouped senses, as all available back-off estimates are known to be wrong in this case. Finally, the sense specific SCFs acquired for each of the different datasets were merged, yielding an SCF distribution specific to a verb form rather than sense.

Two improvements of F-measure can be reported;[12] firstly, an increase in F-measure from 74.3% to 77.6% when the 100% accurate WSD was used to separate the first sense from any other sense (7 verbs); secondly, an increase in F-measure from 75.0% to 78.8% when three sense groups were distinguished (3 verbs). Using this method, we encountered sparse data problems in subcategorization acquisition, since many datasets were simply too small to yield an accurate lexicon. Separating out the data into different datasets not only generated noise but was also unnecessary: the lexicons must be merged in the end, to allow sensible comparison with the baseline system and the use of the extant gold standard data based on verb form.

We therefore refined the method so it does not involve separating data. Instead it involves using back-off estimates specific to the sense distribution in our data, as determined by our WSD system. Thus the method is identical to the baseline method presented above, but a different approach is adopted for constructing back-off estimates: they are now constructed from the back-off estimates of all the senses our WSD system has detected (not just the most frequent), so that the contribution of each set of estimates is weighted according to the frequency of the corresponding senses in corpus data.

We combined the different back-off estimates using linear interpolation. Let $p_j(scf_i), j \in \{1 \ldots n_k\}$ (where n_k is the number of back-off estimates) be the probability of the ith SCF in the jth back-off distribution. The estimated probability of the SCF in the resulting combined back-off distribution is calculated as follows:

$$P(scf_i) = \sum_{j=1}^{n_k} \lambda_j \cdot p_j(scf_i)$$

where the λ_j denote weights for the different distributions and sum to 1. The values for λ_j are determined specific to a verb and are obtained from the probabilistic WSD system, and correspond to the distribution of senses in the examined data (see Section 7.5 for more details on how this information is obtained from a WSD system). Section 7.5 also presents

[11] Note the corpus used for this investigation is SEMCOR, a manually sense annotated corpus and therefore such a split of the data is possible.

[12] Please see Section 7.4.3 for a definition of precision and recall in the context of subcategorization frames.

results obtained from combining our automatic WSD system with the SCF acquisition system, including an investigation of the sensitivity of SCF acquisition to the accuracy of the WSD system employed.

7.4.3 Evaluating SCF Performance

The SCF acquisition system outputs a frequency distribution on SCFs for each verb, which are compared to those in the gold standard SCF distribution for the verb. We therefore need to define some similarity measures for distributions.

We can still determine the precision and recall (and in Section 7.4.2 we reported an increase in F-measure when WSD was used in the SCF acquisition process):

$$\text{precision} = \frac{\text{number of correct SCFs proposed by system}}{\text{total number of SCFs proposed by system}}$$

$$\text{recall} = \frac{\text{number of correct SCFs proposed by system}}{\text{total number of correct SCFs}}$$

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

However, these measures do not take into account the actual frequencies or ranks of the SCFs. Alongside precision and recall, we will also give values for the following six measures which vary in their robustness. The definitions below, taken from Korhonen and Krymolowski (2002), assume $q = \{q_i\}$ and $p = \{p_i\}$ to be the acquired distribution and the gold standard SCF distribution respectively. An individual q_i element denotes the probability of the i^{th} SCF in the distribution q. It is important to report the performance with respect to all these measure as they vary in their sensitivity to noise.

1. **The Intersection measure** (IS) (Lin, 1998)

   $$\text{IS}(p, q) = \frac{2 \times |\text{common}(p, q)|}{|\text{supp}(p)| + |\text{supp}(q)|}$$

   where $\text{supp}(p)$ and $\text{supp}(q)$ are the sets of SCFs with non-zero probability in p and q, and $\text{common}(p, q)$ is the intersection of these two sets.

2. **The Spearman rank correlation coefficient** (RC) (Spearman, 1904) requires the use of ranks r^p and r^q which are found for each SCF separately (using averaged ranks for tied values) in the equation:

   $$\text{RC}(p, q) = \text{corr}(p, q)$$

3. **Cross entropy** (CE) is a measure of the information needed to describe a true distribution p using a model distribution q:

   $$\text{CE}(p, q) = \sum_i -p_i \log(q_i)$$

   In the case when p and q are the same, $\text{CE}(p, q) = H(p)$, the Shannon entropy of p.

4. **The Kullbach-Leibler distance** (KL) is a measure of the additional information needed to describe p using q:

$$\mathrm{KL}(p,q) = \mathrm{CE}(p,q) - H(p) = \sum_i p_i \log\left(\frac{p_i}{q_i}\right)$$

5. **The Jensen-Shannon divergence** (JS):

$$\mathrm{JS}(p,q) = \frac{1}{2}\left[\mathrm{KL}\left(p, \frac{p+q}{2}\right) + \mathrm{KL}\left(q, \frac{p+q}{2}\right)\right]$$

6. **The skew divergence** (SD):

$$\mathrm{SD}(p,q) = \mathrm{KL}(p, \alpha p + (1-\alpha)p)$$

In this work we use $\alpha = 0.99$ since Lee (1999) reported best results with this value.

7.5 Performance of SCF when WSD is used

In this section we find the effect on SCF acquisition performance when a real WSD system (rather than a manual sense tagging) is employed to guide the back-off process as described in Section 7.4.2. We use our WSD system to find an overall sense distribution in the entire corpus and use this distribution to guide the back-off process. By evaluating the resulting SCFs, we investigate the suitability of subcategorization acquisition as a WSD evaluation task.

The 29 high frequency polysemous verbs we use in this evaluation are presented in Table 7.2. For each verb, the table shows how many Levin senses (Levin, 1993) we distinguish; Levin divided up verb senses according to the syntactic constructions they allow, and we map our WordNet senses to this inventory.[13]

The training corpus consists of SEMCOR, the English lexical sample data and the English all words task corpus. Patterns are learnt from 9/10 of the sense tagged data, the remaining 1/10 development corpus being used for finding the smoothing values for the 29 verbs (see Section 8.2). As not all of the 29 verbs appear in the development corpus, default smoothing values are also found.[14]

The test corpus for this task consisted of around 1000 sentences for each verb drawn from the BNC. For each of the 1000 occurrences of the verb, the WSD system produces a probability distribution on senses. The 1000 probability distributions are averaged, to produce an overall probability distribution for the verb, which is used to guide the back-off estimates in subcategorization frame acquisition.

The precision, recall and F-measure results were obtained by comparing the acquired SCF distribution against a manual analysis of the corpus data (Korhonen, 2002). The results are presented in Table 7.4, which also includes performance values for the measures introduced in Section 7.4.3. The table gives the average performance for the baseline systems (no smoothing, and Korhonen's version which smooths with the most frequent

[13]Note that the Levin distinctions will group together semantically similar WordNet senses, and it is these groupings that we have back off distributions for.

[14]The default smoothing values are trained on ambiguous verbs only.

Verb	Num senses	Verb	Num senses
absorb	3	*induce*	2
bear	4	*keep*	3
choose	2	*mark*	3
compose	2	*offer*	2
conceive	2	*proclaim*	2
concentrate	2	*provide*	2
continue	2	*roar*	3
count	3	*seek*	4
descend	2	*settle*	3
distinguish	3	*strike*	3
embrace	2	*submit*	3
establish	3	*wait*	3
find	3	*watch*	2
force	2	*write*	3
grasp	2		

Table 7.2: Test verbs and their senses

sense), and for the SCF system combined with the (26-module) WSD system. Although we have achieved a better performance on the English all words task with a reduced system, it is not clear to us whether the same reduced system would perform better on such a different corpus. A case could be made for dropping the *grammatical relation* module and the *head* modules, since they make use of the RASP parser, which in turn makes use of subcategorization information. However, we believe that there would have to be an iteration step which made use of our results, for the results not to be representative of the effect of WSD on SCF acquisition.

7.5.1 Discussion of Results

Using our WSD system yields 3.3% better F-measure than when the most frequent sense is employed, which in turn yields 6.8% better F-measure than no smoothing. As an extension of this work it would be possible to carry out an experiment to determine how statistically significant this performance increase is. By splitting the input corpora into 10 segments, each containing 100 instances of each target verb, the performance difference between the system using WSD and the baseline system can be compared with the t-test to obtain a statistical significance of the performance increase.

We now discuss the results with respect to the similarity measures introduced in Section 7.4.3. The improvement can be observed on all measures (the only exceptions are precision and the Spearman rank correlation coefficient (RC), which are slightly worse for the most frequent sense than no sense), but particularly on those which evaluate the capability of the system to deal with sparse data. There are 175 unseen gold standard SCFs in the unsmoothed lexicon, 107 are still unseen after smoothing using the most frequent sense, and only 22 remain unseen after WSD is employed.

The effect of WSD is also clear on the more sensitive measures of distributional similarity which consider unfiltered (noisy) SCF distributions and (unlike precision and recall) evaluate the actual frequencies/ranks of SCFs. The Intersection measure (IM) indicates

that there is a large intersection between the acquired and gold standard SCFs when WSD is used (0.97, as opposed to 0.80 with the most frequent sense smoothing). The improvement on RC is smaller (0.04), demonstrating that WSD slightly improves the ranking of SCFs.

From the entropy-based similarity measures (Kullbach-Leibler distance (KL), Cross entropy (CE) and Jensen-Shannon divergence (JS)), KL improves the most when WSD is used in smoothing (0.37 better than using the most frequent sense and 0.56 better than no smoothing). JS, which is considered the most robust of these measures, shows smaller but nevertheless noticeable improvement.

Table 7.3 lists F-measure and JS results for each of the individual test verbs. MF denotes the performance when the most frequent sense is used, and WSD represents the results obtained when we employed our WSD system. We see that, generally, WSD benefits the most those verbs which are highly polysemous with 3-4 senses (e.g. *bear, count, distinguish, roar, wait*) or verbs where various senses differ substantially in terms of subcategorization (e.g. *conceive, continue, embrace, grasp*).

For example, a clear improvement is seen with many of the verbs whose one sense involves mainly NP/PP SCFs such as

He **grasped** *the door's handle, and he entered the chamber of secrets*

and another one involves sentential SCFs like

Does anyone **grasp** *that this was done in 2000, and is old news?*

Due to diathesis alternations (variations in the way verbal-arguments are grammatically expressed), an occurrence of one SCF is likely to give rise to another, related SCF. Thus SCFs tend to occur in data as 'families'. Detection of a verb sense can therefore result in detection of a whole family of new (gold standard) SCFs.

One verb shows worse performance when WSD is used: *seek*. Surprisingly, this verb is highly polysemous and its senses differ substantially in terms of subcategorization. In theory, it is possible that if senses differ a lot in terms of subcategorization and one of them is clearly predominating in the data, then the detection of any of the other senses may result in noise. Our results show, however, that this is not usually the case.

The verbs which do not show (clear) improvement with WSD (e.g. *choose, compose, induce, watch*) are not as highly polysemous (in our coarse grained gold standard), although some of their senses do differ substantially in terms of subcategorization. It is possible that these verbs occurred in our data mostly in their most frequent sense and therefore WSD made little (or no) difference. This is difficult to evaluate without sense disambiguated data.

7.5.2 Suitability of SCF Acquisition to Evaluating WSD

We also investigate how suitable the SCF acquisition task is as an evaluation method for WSD. To judge this, we require a number of WSD systems with various performances. We want to find a correlation between the F-measure of the WSD system (on some gold standard task), and its F-measure on the SCF acquisition task. We have shown in Section 8.2.1 that invoking a smaller number of modules in our WSD system results in different F-measures.

Verb	Senses	F-measure		JS	
		MF	WSD	MF	WSD
absorb	3	40.0%	40.0%	0.08	0.07
bear	4	47.6%	54.6%	0.12	0.10
choose	2	62.5%	62.5%	0.06	0.06
compose	2	50.0%	50.0%	0.10	0.09
conceive	2	38.1%	52.2%	0.11	0.10
concentrate	2	50.0%	50.0%	0.21	0.15
continue	2	48.3%	53.3%	0.06	0.06
count	3	59.3%	64.3%	0.08	0.06
descend	2	61.5%	61.5%	0.03	0.03
distinguish	3	37.5%	47.1%	0.03	0.03
embrace	2	54.6%	61.5%	0.09	0.08
establish	3	23.5%	33.3%	0.04	0.04
find	3	48.0%	48.0%	0.15	0.14
force	2	66.7%	66.7%	0.17	0.16
grasp	2	45.5%	54.6%	0.07	0.05
induce	2	61.5%	61.5%	0.05	0.03
keep	3	50.0%	50.0%	0.14	0.13
mark	3	38.1%	38.1%	0.08	0.08
offer	2	47.6%	47.6%	0.06	0.06
proclaim	2	53.9%	56.0%	0.13	0.10
provide	2	42.9%	42.9%	0.06	0.06
roar	3	69.2%	74.1%	0.11	0.09
seek	4	66.7%	60.0%	0.16	0.12
settle	3	40.0%	46.2%	0.16	0.15
strike	3	61.5%	64.0%	0.16	0.14
submit	3	54.6%	54.6%	0.03	0.02
wait	3	31.6%	47.6%	0.10	0.09
watch	2	48.5%	48.5%	0.19	0.17
write	3	56.3%	60.6%	0.16	0.12

Table 7.3: F-measure and JS for test verbs

Figure 7.2 shows eight combinations of modules along with the F-measure on the SCF acquisition task (each system also invokes the *frequency* and *basic part of speech* modules, thus the third column of the graph corresponds to the most frequent sense baseline performance). The F-measure of the same eight systems was found on the English all words task. The two sets of results correlate with $\rho = 0.97$, showing a very high correlation between gold standard evaluation of WSD systems and SCF acquisition evaluation of WSD systems. Thus we may conclude that the SCF acquisition task is suitable as a task-based task evaluation method for WSD.

7.6 Summary

We focused on task-based methods in this chapter, specifically in the context of SCF acquisition. We introduced Korhonen's (2002) system for SCF acquisition, and we have

Measures	Smoothing method		
	None	Most frequent	WSD
Precision (%)	72.9	72.3	74.6
Recall (%)	31.3	38.9	42.2
F-measure	43.8	50.6	53.9
RC	0.59	0.57	0.61
KL	1.20	0.93	0.56
JS	0.10	0.10	0.09
CE	2.72	2.44	2.30
IM	0.72	0.80	0.97
Unseen SCFs	175	129	22

Table 7.4: Subcategorization acquisition performance

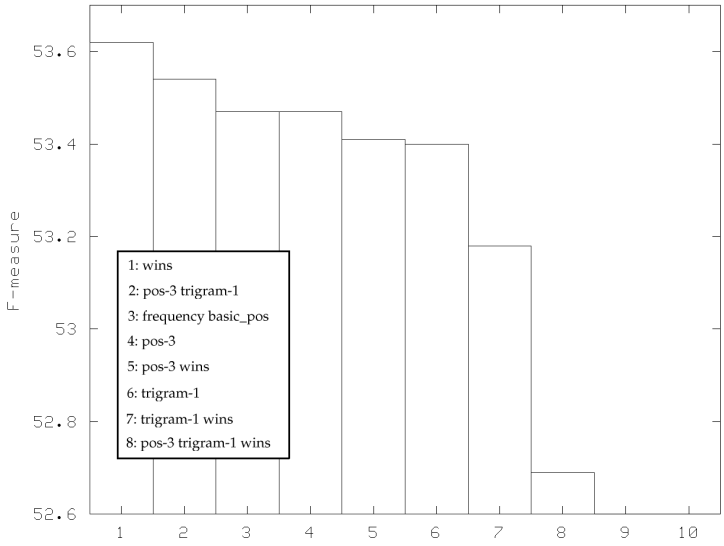

Figure 7.2: Combination of modules for subcategorization acquisition

presented a method for modifying it to benefit from automatic WSD. We introduced similarity measures for SCF distributions, and used our system to investigate the performance of SCF acquisition when this approach to WSD is employed. We conclude that the SCF acquisition task is suitable as a task-based evaluation method for WSD. Evaluating the performance of WSD systems using SCF acquisition has the advantage over a gold standard evaluation that WSD systems need not select precisely the right sense of an arbitrarily fine-grained lexicon.

Chapter 8

Evaluation

8.1 Introduction

This chapter examines the performance of a system that implements the techniques that we presented in Chapters 5 and 6. Three different evaluations are carried out: two gold standard evaluations, and a task-based evaluation. The two gold standard evaluations, on SENSEVAL-2 English tasks, allows us to place our system in context of existing systems. We use the task-based evaluation to investigate how sensitive the subcategorization acquisition task is to WSD, and thus to find out how useful this application is as an evaluation method for WSD.

8.2 English All Words Task

The SENSEVAL-2 corpus consists of 2401 instances taken from running text to be sense tagged. Since no official training data was provided for this task, training examples are not available for some of the words to be annotated. We carry out two stages of training using SEMCOR, and the English lexical sample data as described in Chapter 5 (Section 5.4):

1. Patterns are acquired from 9/10 of the training corpora for each of the 26 modules (the *frequency* and *basic part of speech* modules are not trained).

2. The remaining 1/10 of the training corpus is used to find the optimal smoothing values for each module (using the full 26 module system), for 60 of the most frequent words in the test corpus. The number of words to individually train was chosen based on the following principles: the words had to appear in the training corpus at least three times, and, if this gave rise to too many words, the total number of words chosen had to be trainable within a reasonable amount of time (where one word takes about 25 minutes to train on a Pentium 4 2400MHz machine).

We present the results in Table 8.1 for the three methods of combining modules: the linear interpolation, the Dempster-Shafer and the Bayes Rule combinations. The baseline reflects the performance of the most frequent sense. The table contains precision and recall, as well as the F-measure (defined in Section 7.2.1).

Results are shown for two types of evaluation: the forced choice evaluation, in which only one sense tag is assigned to sense tagged words (the sense tag corresponding to the

Combination	Evaluation	Precision	Recall	F-measure
Linear Interpolation	Forced	59.2%	62.1%	60.6%
	Log odds	59.3%	62.1%	60.7%
Dempster-Shafer	Forced	63.1%	62.1%	62.6%
	Log odds	63.2%	62.2%	62.7%
Bayes Rule	Forced	64.1%	63.0%	63.6%
	Log odds	64.1%	63.1%	63.6%
N/A	Baseline	61.0%	60.0%	60.5%

Table 8.1: Results for the English all words task

highest probability), and the log odds ratio evaluation (Dagan and Itai, 1994), in which all the senses (with probability p) satisfying

$$log\left(\frac{\text{highest probability}}{p}\right) \geq T$$

(where T is a threshold value) are returned and their probabilities are normalized. The log odds ratio allows us to return a number of senses if their probability is close to that of the most probable sense (the sense with the highest probability). We investigated the value of T, by varying it between 0.002 and 0.4 in steps of 0.002. The optimal value was found to be 0.1, which generated the results in Table 8.1. However, the results did not deviate much from those given with any of the tested thresholds (for the Bayes Rule combination, the lowest F-measure was only 0.05% lower than the maximum, and occurred when senses with probabilities up to 0.002 lower than the highest probability were included).

Given the evaluation method in SENSEVAL, it is not reasonable to score the full probability distribution. Even when the system is very confident of the correct sense assignment, smoothing will ensure that some probability mass will be assigned to the remaining senses. In some applications this may be considered a benefit, but with the current evaluation method it degrades performance.

Table 8.1 also includes the performance of the baseline (the most frequent sense heuristic), which is lower than the perfect baseline presented in Table 2.6. This is to be expected, as the baseline performance is dependent on the performance of the tagger, correct identification of multiwords[1] and assignments of the "U" tag.

Both the Dempster-Shafer and the Bayes Rule combination methods outperform the linear interpolation combination. However, interestingly, even though the Bayes Rule combination method eliminates a bias towards the most frequent sense, it does not lead to a great improvement over the Dempster-Shafer combination method. This may be due to the contribution of the *frequency* module; the distributions produced by the remaining modules rarely separate the senses as clearly. Nevertheless, using the t-test it is possible to calculate that the Bayes Rule combination is significantly better than the Dempster-Shafer combination (95% confidence), and the Dempster-Shafer combination is in turn significantly better than the linear interpolation combination (99% confidence).

[1]Words which form part of a multiword can be joined together using "_" and these (now one word) forms appear in WordNet. However, we do not preprocess multiwords, as these did not appear to be consistently annotated in the gold standard.

These results can be directly compared with the English all words task results in Table 2.6, to give our system a ranking among current state-of-the-art systems. The full system containing 26 modules would rank second (when comparing F-measure, with Bayes Rule combination). With an F-measure of 63.6%, it comes after the SMUaw system.

8.2.1 Choosing the Right Modules

We investigated the performance of our system with a lower number of modules; there are a number of successful systems using a subset of the information sources we exploit in our full system. As an exhaustive search is not possible (26 modules give rise to 2^{26} possible module combinations), we must restrict their number in a different way.[2] We can judge the usefulness of a module using the smoothing value produced for it – the smoothing value was chosen to maximize F-measure on the development corpus, so the lower the smoothing value, the more accurate (and therefore useful) a module is. For example, the smoothing value for the *head0* module (identifying the target word as a head of an NP or VP) is 772, and so this module mostly outputs probabilities close to the uniform distribution. A summary of all smoothing values for words which are not individually trained in the English all words task when the Bayes Rule combination method is employed can be seen in Table 8.2.

Module	Value	Module	Value	Module	Value
Frequency	1.00	*Window*	0.91	*Lemma2*	0.53
Basic pos	0.02	*Trigram-1*	0.81	*Lemma3*	0.53
Pos-3	0.55	*Trigram0*	0.72	*Head-2*	32.0
Pos-2	0.99	*Trigram1*	0.95	*Head-1*	0.75
Pos-1	0.77	*Lemma-3*	0.51	*Head0*	115
Pos-0	3.3	*Lemma-2*	0.59	*Head1*	5.00
Pos1	0.86	*Lemma-1*	0.86	*Head2*	28.0
Pos2	0.83	*Lemma0*	3.20	*GRs*	267
Pos3	0.84	*Lemma1*	0.94		

Table 8.2: Smoothing values for the English all words task

For the optimization, the program was always invoked with the *frequency* and *basic part of speech* modules, and used Bayes Rule combination. We selected the top 7 modules:[3] *head-1*, *trigrams0*, *lemma-3*, *lemma-2*, *trigrams-1*, *pos-1*, and *pos-3*. A graph depicting the F-measure when various module combinations are invoked is shown in Figure 8.1. The performance of the baseline (*frequency* and *basic pos*) is shown in column 10, and other columns report the F-measure obtained from combining the *frequency* and *basic pos* modules with those specified in the key on the graph.

The figure shows that it is possible for our system to improve on its performance (reaching the F-measure of 65.3%). The increase in performance when modules are dropped may seem surprising, however there are two possible reasons for this:

[2]Methods such as simulated annealing often reach a local optimum and therefore were considered unsuitable for this experiment.

[3]The number of modules was chosen so that the whole optimization did not run for too long. Note that the WSD system program was not written with an emphasis on efficiency.

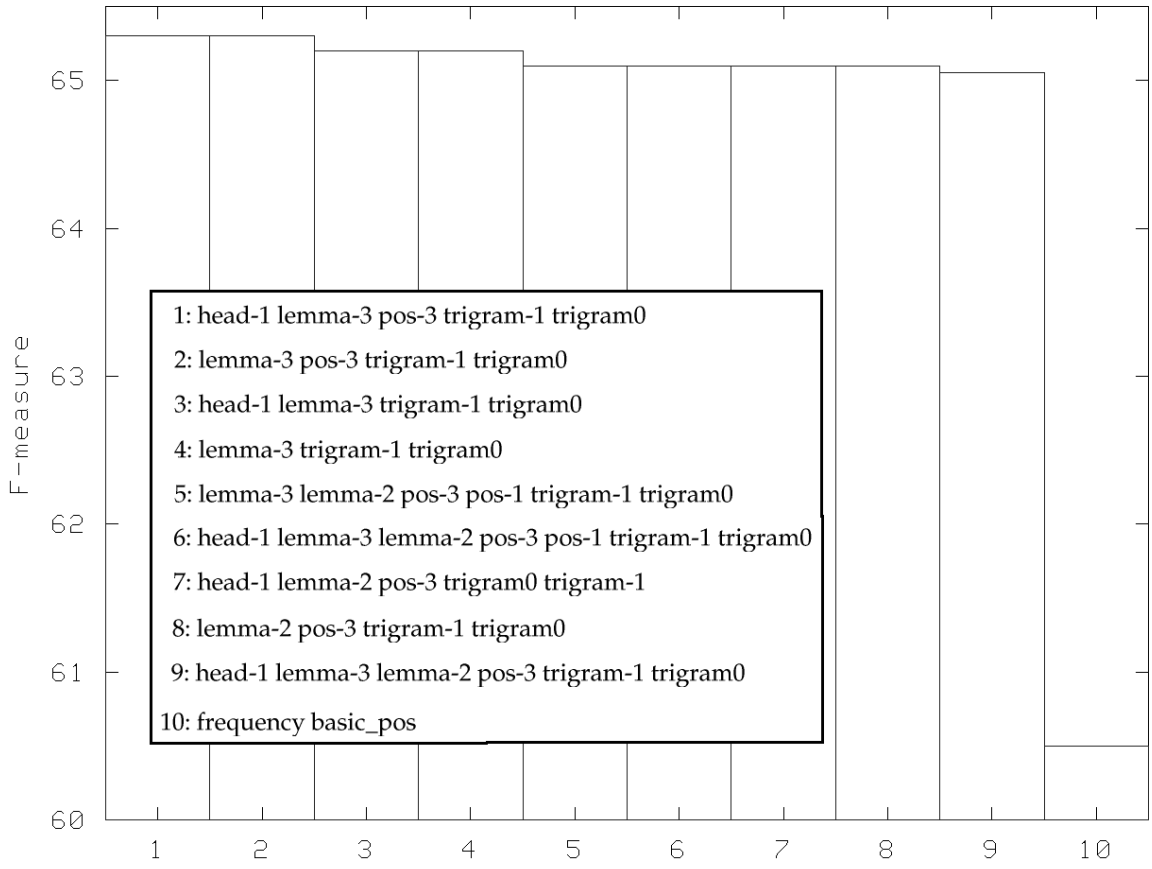

Figure 8.1: Combination of modules for English all words task

1. Some modules encode very similar information (e.g., *pos1* and *head1* will both tell us if the following word is a noun), which may be getting over-emphasized (when wrong) when the modules are invoked together so removing them from the combination may result in an increase in performance. This is an instance of a violation of the assumed independence of modules.

2. The smoothing value may be high for some modules, making the modules approximate the uniform distribution, but the modules may still be inaccurate and thus have a deleterious effect.

8.2.2 Error Analysis

It is useful to look at common errors made by the system, to examine its strengths and weaknesses when run on real data. We looked at the first 100 errors made on the English all words task by the full 26 module system using Naive Bayes combination method.

67% of the errors were due to the system preferring the most frequent sense over the correct sense because of a lack of relevant training data. The output of each trained

module is the uniform distribution if no training data is present, and so the 26 module system collapses to the most frequent sense baseline system.

A further 10% of the errors were due to the tagger attaching a high probability to an incorrect part of speech. This frequently occurred with a noun/adjective distinction (e.g., consider the word *english*) where the tagger assigned an almost equal probability to both parts of speech. If there was also no training data present, the system will usually select a sense from the part of speech with fewest senses. For an example, see Table 8.3 where the noun senses (denoted by %1) are assigned a probability of 0.494 and the adjective sense (%3) has a probability of 0.487.

Sense	Tagger	Frequency	Tagger $*$ Frequency
english%1:10:00::	0.493706	0.647059	0.319457
english%1:09:00::	0.493706	0.147059	0.072604
english%1:18:00::	0.493706	0.176471	0.087125
english%1:11:00::	0.493706	0.029412	0.014521
english%3:01:00::	0.487426	1.000000	0.487426
none	0.018868	1.000000	0.018868

Table 8.3: An example tagger error

The remaining 23% of the errors cannot be attributed to any major cause, but are rather a collection of rare causes. For example, the system does not handle multiwords, and so these are all scored as incorrect, but there are only 70 instances of multiwords in the test corpus out of 2473, less than 3%. As a second example, there is a mismatch between the lemmatizer and WordNet, causing words such as *don't* to be broken up in such a way that they would never be scored correctly.

8.3 English Lexical Sample

In Chapter 3, we have presented our work on the effect of increasing the amount of training data on performance of our WSD system. We used the *line* corpus (Leacock et al., 1993), which contains 4148 instances of the word *line* each annotated with one of six senses. We found the average performance (F-measure) to rise sharply up to 300 training instances after which the gradient begun to level off. A performance plateau for the six sense word *line* was reached at about 500 instances.

In the SENSEVAL-2 English lexical sample task, the number of instances tagged for each word was $75 + 15n$ where n is the number of senses a word has within the chosen part of speech (Kilgarriff, 2002; Palmer et al., 2002)). Two thirds of the total annotated data for each word was used for training. The average number of training instances in the English lexical sample task training corpus is 121, and 93% of the words have fewer than 200 instances. The average polysemy for the English lexical sample task was 9.1 (rather than the 5.4 average for the English all words task), it is not clear whether the English lexical sample gives a fair representation of the system's performance as our small experiment with the word *line* indicated the need for many more training examples. The performance of the full 26 module system trained on the English lexical sample training corpus is presented in Table 8.4.

Combination	Evaluation	Precision	Recall	F-measure
Linear Interpolation	Forced	47.5%	47.8%	47.6%
Dempster-Shafer	Forced	53.8%	53.8%	53.8%
Bayes Rule	Forced	54.3%	54.3%	54.3%
N/A	Baseline	39.4%	38.7%	39.1%

Table 8.4: Results for the English lexical sample task

8.4 Summary

We have presented the results of our WSD system obtained using a gold standard evaluation, namely the SENSEVAL-2 English all words task. The full system would rank second compared to other systems participating in the English all words task. We also present results for the SENSEVAL-2 English lexical sample task, where the system appears to suffer slightly due to the small amount of training data available.

Chapter 9

Conclusions

The aim of this work has been to show that probabilistic word sense disambiguation systems are strong competitors to current, often less theoretically motivated, state-of-the-art WSD systems.

Chapter 2 surveys existing approaches to WSD, and examines their performance in the systems submitted to SENSEVAL-2. We describe WSD resources including an investigation of the performance variation with an increased training corpus in Chapter 3. The design of our new probabilistic WSD system is presented in Chapter 5. The main feature of the system is that it is composed of multiple probabilistic components: such modularity is made possible by an application of Dempster-Shafer theory, Bayes Rule and Lidstone's smoothing method. In Chapter 8 we evaluate our system on the SENSEVAL-2 English all words task: in its raw form it appears second in the list of results. We also show that the subcategorization frame acquisition task is suitable for evaluating WSD systems.

This work has made a number of original contributions to knowledge; these were pointed out throughout the text, and here we collect them together in categories and summarize them:

Word Sense Disambiguation: The success of our probabilistic WSD system has demonstrated the effectiveness of probabilistic methods in WSD; this trend can also be observed in other areas of NLP such as statistical parsing, or Ge et al.'s (1998) anaphora resolution algorithm which combined a number of anaphora resolution approaches using probabilistic methods. To create our WSD system, we adapted a number of existing WSD approaches to function as probabilistic modules, and we have given a precise description of these probabilistic modules for others to use in future systems.

Task-based evaluations are becoming more popular in NLP, being an absolute measure of a system's performance on a given task. Experiments with our probabilistic WSD system give an extremely high correlation between subcategorization frame acquisition performance and WSD performance, thus demonstrating the suitability of SCF acquisition as a WSD evaluation task.

The SENSEVAL competition provides a common evaluation framework for comparing the performance of WSD systems. Our analysis of the results for each system in the SENSEVAL-2 English all words task uncovered hidden correlations between system performance and syntactic features, and in Chapter 4 we use decision trees to examine system specializations and develop a combination WSD system.

Statistical NLP: Dempster-Shafer theory uses notions of belief and plausibility to combine probabilistic information within a common belief framework. Our work has demonstrated an application of the theory to combining probability distributions on word senses, thereby adding another technique to statistical NLP.

Our implementation of Lidstone's smoothing makes the following contributions: it provides a uniform mechanism for weighting modules based on their accuracy, removing the need for an additional confidence weighting scheme; and smoothing values are found for individual words where possible, giving an extra degree of specialization.

Subcategorization Frame Acquisition: SCF acquisition is a useful aid to parsing, providing vital information about verb arguments. We have demonstrated that it can be improved using WSD, and even that an imperfect WSD system can lead to a performance increase.

9.1 Future Work

We list a number of possible improvements and extensions that could be made to our work:

- Our current smoothing method uses a function of accuracy to smooth probability distributions. Mathematical analysis combined with empirical investigation may uncover the optimum function of accuracy for our application.

- We have argued that the performance of our WSD system may be increased when a larger training corpus is used. A corpus could be created based on Mihalcea and Moldovan's GENCOR. This is an automatically acquired corpus, created using bootstrapping techniques seeded with SEMCOR and WordNet examples, and using internet queries to build up instances of tagged words. Preliminary investigations have indicated that a corpus generated in this manner is rather unbalanced: e.g., many instances are produced for senses which are rare but have monosemous synonyms. However, it may be possible to artificially simulated a balanced corpus.

- A thorough investigation of dependence between modules could be carried out to understand their relatedness and to improve performance. For example, this could be done using covariance matrices from statistics.

- Initial experiments have shown that an increase in WSD performance may be gained by combining WSD with anaphora resolution (McCarthy et al., 2002). Resolving third person pronouns (such as *it*) to their antecedents provides more contextual information for sense tagging the antecedent and words near the pronoun, possibly increasing the accuracy of the WSD system.

Bibliography

ALPAC. Language and machine computers in translation and linguistics. Technical report, National Research Council Automatic Language Processing Advisory Committee, Washington, D.C., 1966.

S. Atkins. Tools for corpus-aided lexicography: the HECTOR project. *Acta Linguistica Hungarica*, 41:5–72, 1992–93.

M. Banko and E. Brill. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of 39th Annual Meeting of the ACL*, pages 26–33, 2001.

Y. Bar-Hillel. The present status of automatic translation of languages. In F. Alt, A. Booth, and R. Meagher, editors, *Advances in Computers*, pages 91–163. Academin Press, New York, 1960.

Y. Bar-Hillel. The outlook for computational semantics. In *Proceedings of the Conference on Computer-Related Semantic Analysis*, pages I/1–14, 1965.

A. Bendjebbour, Y. Delignon, L. Fouque, V. Samson, and W. Pieczynski. Multisenson image segmentation using dempster-shafer fusion in markov fields context. *IEEE Transactions on Geoscience and Remote Sensing*, 39(8), 2001.

B. K. Boguraev and E. J. Briscoe. Large lexicons for natural language processing utilising the grammar coding system of the *Longman Dictionary of Contemporary English*. *Computational Linguistics*, 13(4):219–240, 1987.

E. J. Briscoe and J. Carroll. Robust accurate statistical annotation of general text. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1499–1504, 2002.

E. J. Briscoe and J. Carroll. Automatic extraction of subcategorization from corpora. In *Proceedings of ACL ANLP97*, pages 356–363, 1997.

P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, J. Lafferty, R. Mercer, and P. Rossin. A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85, 1990.

P. Brown, S. Della Pietra, V. Della Pietra, and R. Mercer. Word sense disambiguation using statistical methods. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, pages 264–304, 1991.

R. Bruce and J. Wiebe. Word-sense disambiguation using decomposable models. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pages 139–145, 1994.

J. Carroll, E. Briscoe, and A. Sanfilippo. Parser evaluation: A survey and a new proposal. In *Proceedings of the International Conference on Language Resources and Evaluation*, pages 447–454, 1998a.

J. Carroll, G. Minnen, and E. Briscoe. Can subcategorisation probabilities help a statistical parser? In *Proceedings of the 6th ACL/SIGDAT Workshop on Very Large Corpora*, pages 118–126, 1998b.

E. Charniak. *Statistical Language Learning*. Cambridge: MIT Press, 1993.

S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 310–318, 1996.

Y. Choueka and S. Lusignan. Disambiguation by short contexts. *Computer and the Humanities*, 19:22–29, 1985.

A. Copestake and T. Briscoe. Semi-productive polysemy and sense extension. *Journal of Semantics*, 12:15–67, 1995.

I. Dagan and A. Itai. Word sense disambiguation using a second language monolingual corpus. *Computational Linguistics*, 20:563–596, 1994.

A. P. Dempster. Upper and lower probabilities induced by a multivalued mapping. *Annals of Mathematical Statistics*, 39:325–339, 1967.

T. G. Dietterich and E. B. Kong. Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. Technical report, Department of Computer Science, Oregon State University, 1995.

P. Edmonds and A. Kilgarriff. Introduction to the special issue on evaluating word sense disambiguation systems. *Journal of Natural Language Engineering*, 8(4):279–291, 2002.

D. Elworthy. Does Baum-Welch re-estimation help taggers? In *Proceedings of the 4th Conference on Applied NLP*, pages 53–58, 1994.

R. Florian and D. Yarowsky. Modeling consensus: Classifier combination for word sense disambiguation. In *Proceedings of EMNLP'02*, pages 25–32, 2002.

W. Gale, K. Church, and D. Yarowsky. One sense per discourse. In *Proceedings of the 4th DARPA Speech and Natural Language Workshop*, pages 233–237, 1992a.

W. Gale, K. W. Church, and D. Yarowsky. Estimating upper and lower bounds on the performance of word-sense disambiguation programs. In *Proceedings, 30th ACL*, pages 249–256, 1992b.

W. Gale, K. W. Church, and D. Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26(5–6):415–439, 1992c.

N. Ge, J. Hale, and E. Charniak. A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora*, pages 161–170, 1998.

I. J. Good. The population frequencies of species and the distribution of population parameters. *Biometrika*, 40(3/4):237–264, 1953.

R. Grishman, C. Macleod, and A. Meyers. Comlex syntax: building a computational lexicon. In *International Conference on Computational Linguistics, COLING-94*, pages 268–272, 1994.

P. J. Hayes. A process to implement some word-sense disambiguation. Technical report, Institut pour les Etudes Sémantiques et Cognitives, Université de Genève, 1976.

M. Hearst. Noun homograph disambiguation using local context in large corpora. In *Proceedings of the 7th Annual Conference of the University of Waterloo Centre for the New Oxford English Dictionary*, pages 1–22, 1991.

D. Hindle. Noun classification from predicate-argument structures. In *Proceedings of the 28th Annual Meeting of the ACL*, pages 268–75, 1990.

V. Hoste, I. Hendrickx, W. Daelemans, and A. Van Den Bosch. Parameter optimization for machine-learning of word sense disambiguation. *Journal of Natural Language Engineering*, 8(4):311–325, 2002.

N. Ide and J. Véronis. The state of the art. *Computational Linguistics*, 24:1–40, 1998. Introduction to the Special Issue on Word Sense Disambiguation.

A. Jameson. Numerical uncertainty management in user and student modeling: An overview of systems and issues. *User Modelling and User-Adapted Interaction*, 5(3/4): 193–251, 1995.

D. Jurafsky and J. Martin. *Speech and Language Processing*. Prentice Hall, 2000.

J. J. Katz and J. A. Fodor. The structure of a semantic theory. In J. J. Katz and J. A. Fodor, editors, *The structure of language*, pages 479–518. Prentice Hall, 1964.

A. Kilgarriff. English lexical sample task description. In Preiss and Yarowsky (2002), pages 17–20.

A. Kilgarriff. "I don't believe in word senses". *Computers and the Humanities*, 31(2): 91–113, 1997.

A. Kilgarriff. SENSEVAL: An exercise in evaluating word sense disambiguation programs. In *Proceedings of LREC*, pages 581–588, 1998.

A. Kilgarriff and J. Rosenzweig. Framework and results for English SENSEVAL. *Computers and the Humanities*, 34(1–2):15–48, 2000.

R. Koeling, D. McCarthy, and J. Carroll. Domain-specific sense distributions and predominant sense acquisition. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 419–426, 2005.

A. Korhonen. *Subcategorization Acquisition.* PhD thesis, University of Cambridge, 2002.

A. Korhonen and Y. Krymolowski. On the robustness of entropy-based similarity measures in evaluation of subcategorization acquisition systems. In *Proceedings of the 6th Conference on Natural Language Learning*, pages 91–97, 2002.

A. Korhonen and J. Preiss. Improving subcategorization acquisition using word sense disambiguation. In *Proceedings of ACL*, pages 48–55, 2003.

R. Krovetz. More than one sense per discourse. In *Proceedings of the ACL-SIGLEX Workshop*, 1998.

S. Kurohashi. SENSEVAL-2 Japanese translation task. In Preiss and Yarowsky (2002), pages 37–40.

C. Leacock, G. Towell, and E. Voorhees. Corpus-based statistical sense resolution. In *Proceedings of the ARPA Workshop on Human Language Technology*, pages 260–265, 1993.

C. Leacock, M. Chodorow, and M. A. Miller. Using corpus statistics and WordNet relations for sense identification. *Computational Linguistics*, 24(1):147–165, 1998.

L. Lee. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the ACL*, pages 25–32, 1999.

G. Leech. 100 million words of English: the British National Corpus. *Language Research*, 28(1):1–13, 1992.

M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of AGM SIGDOC Conference*, pages 24–26, 1986.

B. Levin. *English Verb Classes and Alternations.* Chicago University Press, 1993.

H. Li and N. Abe. Generalizing case frames using a thesaurus and the mdl principle. *Computational Linguistics*, 24(2):217–244, 1998.

D. Lin. Automatic retrieval and clustering of similar words. In *Proceedings of the COLING-ACL'98*, pages 768–773, 1998.

S. Madhu and D. W. Lytle. A figure of merit technique for the resolution of non-grammatical amiguity. *Mechanical Translation*, 8(2):9–13, 1965.

C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.

M. Masterman. Semantic message detection for machine translation, using an interlingua. In *Proceedings of 1961 International Conference on Machine Translation of Languages and Applied Language Analysts*, pages 437–475, 1961.

D. McCarthy. Disambiguating nouns, verbs and adjectives using automatically acquired selectional preferences. *Computational Linguistics*, 29(4):639–654, 2003.

D. McCarthy, J. Carroll, and J. Preiss. Disambiguating noun and verb senses using automatically acquired selectional preferences. In Preiss and Yarowsky (2002), pages 119–122.

R. Mihalcea. Word sense disambiguation using pattern learning and automatic feature selection. *Journal of Natural Language and Engineering*, 8(4):343–358, 2002.

R. Mihalcea. Co-training and self-training for word sense disambiguation. In *Proceedings of the Conference on Natural Language Learning*, pages 33–40, 2004.

R. Mihalcea and T. Chklowski, editors. *Proceedings of SENSEVAL-3: Third International Workshop on Evaluating Word Sense Disambiguating Systems*, 2004.

R. Mihalcea and D. I. Moldovan. A highly accurate bootstrapping algorithm for word sense disambiguation. *International Journal on Artificial Intelligence Tools*, 10(1–2), 2001.

R. Mihalcea and D. I. Moldovan. Pattern learning and active feature selection for word sense disambiguation. In Preiss and Yarowsky (2002), pages 127–130.

R. Mihalcea and D. I. Moldovan. An automatic method for generating sense tagged corpora. In *Proceedings of AAAI-99*, pages 461–466, 1999.

G. Miller, R. Beckwith, C. Felbaum, D. Gross, and K. Miller. Introduction to WordNet: An on-line lexical database. *Journal of Lexicography*, 3(4):235–244, 1990.

T. M. Mitchell. *Machine Learning*. McGraw-Hill International Editions, 1997.

R. J. Mooney. Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP-96)*, pages 82–91, 1996.

H. Ng and H. Lee. Integrating multiple knowledge sources to disambiguate word sense. an exemplar-based approach. In *Proceedings of the 34th Meeting of the Association for Computational Linguistics (ACL-96)*, pages 40–47, 1996.

H. T. Ng, B. Wang, and Y. S. Chan. Exploiting parallel texts for word sense disambiguation: An empirical study. In *Proceedings of the Association for Computational Linguistics*, pages 455–462, 2003.

M. Palmer, C. Fellbaum, S. Cotton, L. Delfs, and H. T. Dang. English tasks: All-words and verb lexical sample. In Preiss and Yarowsky (2002), pages 21–24.

J. Pearsall, editor. *New Oxford Dictionary of English*. Oxford: Clarendon Press, 1998.

T. Pedersen. A simple approach for building ensembles of naive Bayesian learners for word sense disambiguation. In *Proceedings of ANLP–NAACL 2000*, pages 63–69, 2000.

T. Pedersen. A decision tree of bigrams is an accurate predictor of word senses. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-01)*, pages 79–86, 2001.

T. Pedersen. Machine learning with lexical features: The Duluth approach to Senseval-2. In Preiss and Yarowsky (2002), pages 139–142.

M. P. Perrone and L. N. Cooper. When networks disagree: Ensemble methods for hybrid neural networks. In R. J. Mammone, editor, *Neural Networks for Speech and Image Processing*, pages 126–142. Chapman-Hall, 1993.

V. Poznański. A relevance-based utterance processing system. Technical report, University of Cambridge, 1992.

J. Preiss and A. Korhonen. Improving subcategorization acquisition with WSD. In *Proceedings of the Word Sense Disambiguation Workshop*, pages 102–108, 2002.

J. Preiss and D. Yarowsky, editors. *Proceedings of SENSEVAL-2: Second International Workshop on Evaluating Word Sense Disambiguating Systems*, 2002.

J. Preiss, A. Korhonen, and E. J. Briscoe. Subcategorization acquisition as an evaluation method for WSD. In *Proceedings of LREC*, pages 1551–1556, 2002.

P. Procter, editor. *Cambridge International Dictionary of English*. Cambridge University Press, 1995.

J. Pustejovsky. *The Generative Lexicon*. MIT Press, 1995.

M. R. Quillian. A semantic coding technique for mechanical english paraphrasing. Technical report, Mechanical translation Group, Research Laboratory of Electronics, M.I.T, 1962.

M. R. Quillian. The teachable language comprehender: A simulation program and theory of language. *Communications of the ACM*, 12(8):469–476, 1969.

J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman Publishers, 1993.

P. Resnik. Wordnet and distributional analysis: a class-based approach to lexical discovery. In *Worshop Notes, Statistically-Based NLP Techniques AAAI*, pages 54–64, 1992.

P. Resnik and D. Yarowsky. A perspective on word sense disambiguation methods and their evaluation. In *Proceedings of the SIGLEX Workshop Tagging Text with Lexical Semantics*, pages 79–86, 1997.

P. Resnik and D. Yarowsky. Distinguishing systems and distinguishing senses: New evaluation methods for word sense disambiguation. *Natural Language Engineering*, 5(2): 113–134, 1999.

P. M. Roget. *Roget's International Thesaurus*. New York: Thomas Y. Crowell, 1946.

D. Roland and D. Jurafsky. Verb sense and verb subcategorization probabilities. In S. Stevenson and P. Merlo, editors, *The Lexical Basis of Sentence Processing: Formal, Computational, and Experimental Issue*. Cambridge University Press, Jon Benjamins, Amsterdam, 2001. To appear.

D. Roland, D. Jurafsky, L. Menn, S. Gahl, E. Elder, and C. Riddoch. Verb subcategorization frequency differences between business-news and balanced corpora. In *ACL Workshop on Comparing Corpora*, pages 28–34, 2000.

M. Sanderson and C. J. Van Rijsbergen. The impact on retrieval effectiveness of skewed frequency distributions. *ACM Transactions on Information Systems*, 17(4):440–465, 1999.

A. Sarkar and D. Zeman. Automatic extraction of subcategorization frames for Czech. In *19th International Conference on Computational Linguistics*, pages 691–697, 2000.

G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

S. Small. *Word Expert Parsing: A Theory of Distributed Word-based Natural Language Understanding*. PhD thesis, Department of Computer Science, University of Maryland, 1980.

S. Small and C. Rieger. Parsing and comprehending with word experts (a theory and its realisation). In W. Lehnert and M. Ringle, editors, *Strategies for Natural Language Processing*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1982.

K. Sparck Jones and J. Galliers. *Evaluating Natural Language Processing Systems, an Analysis and Overview*. Springer Verlag, lecture Notes in Artificial Intelligence 1083, 1996.

C. Spearman. The proof and measurement of association between two things. *Americal Journal of Psychology*, 15:72–101, 1904.

M. Stevenson. *Word Sense Disambiguation: The Case for Combining Knowledge Sources*. CSLI Publications, Stanford, CA., 2003.

M. Stevenson. *Multiple Knowledge Sources for Word Sense Disambiguation*. PhD thesis, University of Sheffield, 1999.

M. Stevenson and Y. Wilks. The interaction of knowledge sources in word sense disambiguation. *Computational Linguistics*, 27(3):321–349, 2001.

M. Stevenson and Y. Wilks. Combining weak knowledge sources for sense disambiguation. In *Proceedings of the International Joint Conference for Artificial Intelligence (IJCAI-99)*, pages 884–889, 1999.

J. Veenstra, A. van den Bosch, S. Buchholz, W. Daelemans, and J. Zavrel. Memory-based word sense disambiguation. *Computers and the Humanities*, 34(1–2):171–177, 2000.

J. Véronis and N. Ide. Word sense disambiguation with very large neural networks extracted from machine readable dictionaries. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING-90)*, pages 389–394, 1990.

W. Weaver. Translation. In W. N. Locke and A. D. Booth, editors, *Machine translation of languages*, pages 15–23. John Wiley and Sons, New York, 1955. Originally Mimeographed, 12pp, 1949.

Y. Wilks. On-line semantic analysis of english texts. *Mechanical Translation*, 11(3–4): 59–72, 1968.

Y. Wilks. *Grammar, Meaning and the Machine Analysis of Language*. Routledge, London, 1972.

Y. Wilks and M. Stevenson. The grammar of sense: Is word sense tagging much more than part-of-speech tagging? Technical Report CS-96-05, University of Sheffield, 1996.

I. H. Witten and E. Frank. *Data mining: Practical Machine Learning Tools and Techniques with Java Implementations*, chapter 8. Morgan Kaufmann Publishers, 2000.

L. Xu, A. Krzyzak, and C. Y. Suen. Several methods for combining multiple classifiers and their applications in handwritten character recognition. *IEEE Trans. on System, Man and Cybernetics*, 22(3):418–435, 1992.

D. Yarowsky. Hierarchical decision lists for word sense disambiguation. *Computers and the Humanities*, 34(1/2):179–186, 2000.

D. Yarowsky. Decision lists for lexical ambiguity resolution: Application to accent restration in spanish and french. In *Proceedings of ACL*, pages 88–95, 1994.

D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Annual Meeting of the Association of Computational Linguistics*, volume 33, pages 189–196, 1995.

D. Yarowsky. *Three Machine Learning Algorithms for Lexical Ambiguity Resolution*. PhD thesis, University of Pennsylvania, 1996.

D. Yarowsky and R. Florian. Evaluating sense disambiguation across diverse parameter spaces. *Journal of Natural Language Engineering*, 8(4):293–310, 2002.

G. K. Zipf. *Human Behaviour and the Principle of Least-Effort*. Addison-Wesley, Cambridge MA, 1949.

A. M. Zwicky and J. M. Sadock. Ambiguity tests and how to fail them. *Syntax and Semantics*, 4:1–36, 1975.