Euskal Herriko Unibertsitatea/ Universidad del País Vasco

Lengoaia eta Sistema Informatikoak Saila
Departamento de Lenguajes y Sistemas Informáticos

Supervised Word Sense Disambiguation: Facing Current Challenges

David Martinez Iraolak

Informatikan Doktore titulua eskuratzeko aurkezturiko

Tesia

Donostia, 2004ko urria

Esker onak / Acknowledgements

Lehenbizi, Ixa taldeko kideen laguntza eskertu nahi dut. Ixa taldeak eskaini dizkit ikerkuntzarako beharrezkoak diren ingurunea eta baliabide guztiak. Gure lanean erabili ditugun euskara lantzeko tresnak hemen garatuak izan dira.

Bereziki Eneko aipatu nahi dut, hemen aurkezten den ikerketa elkarlanean egin baitugu. Oierren laguntza ere eskertu nahi dut, internetetik corpus erauzketan egin duen lana oso erabilgarria izan baita tesi honetarako. Aitorrek, berriz, liburuxka hau latex-en egiterakoan bizitza erraztu dit, eta eskerrak eman nahi dizkiot. Baita UPCko (Universitat Politecnica de Catalunya) Lluisi ere, proiektu honetan bere laguntza eskaini baitit AdaBoost algoritmoarekin lan egitean.

First of all, I want to thank the help of my friends in the Ixa group. The Ixa group has provided the necessary environment and resources for this research. The Basque-processing tools that we have used in our work have been developed here.

I would like to mention specially Eneko, because we have done this research-work jointly. I also want to thank Oier, because his help extracting corpora from the internet has been very helpful in this thesis. Aitor, too, has saved me a lot of time when editing in latex. Lluis, from UPC (Universitat Politecnica de Catalunya), has also collaborated in this project when we worked with the AdaBoost technique.

For much of our work, we relied on software publicly available for research. We wish to thank the developers for their effort and their contribution to push forward this and other research fields. Different NLP groups also contributed to this work by providing us insight in the problem, and their resources and

tools for research.

I would like to thank John Carroll and Diana McCarthy, from the University of Sussex, who helped us at different points of this dissertation. John Carroll provided the results of his parser on our data, which we used to compare it with other publicly available parsers. Diana McCarthy applied her method for ranking senses, which we applied in our research.

I would like to thank David Yarowsky's group, from Johns Hopkins University, for a nice stay in the fall of 2002. They gave me ideas and resources that I have used in this dissertation. Gerard Escudero's group, from UPC, has also worked with us, sharing tools and experiences. I would like to mention specially German, now in the Ixa group, for his close collaboration during this work.

This research has been partially funded by a research grant from the Basque Government (grant AE-BFI:01.245), and by the European Commission (MEANING IST-2001-34460).

Contents

Acronyms and abbreviations

ACL: Association for Computational Linguistics
AI: Artificial Intelligence
BC: Brown Corpus
BNC: British National Corpus
DL: Decision Lists
DSO: Defense Science Organization
GR: Grammatical Relations
HMM: Hidden Markov Models
IBL: Instance Based Learning
IGR: Instantiated Grammatical Relations
MBL: Memory Based Learning
ME: Maximum Entropy
MFS: Most Frequent Sense
ML: Machine Learning
MLE: Maximum Likelihood Estimation
MM: Mixture Model
MT: Machine Translation
MUC: Message Understanding Conference
NB: Naive Bayes
NE: Named Entity
NLP: Natural Language Processing
NLU: Natural Language Understanding
PoS: Part of Speech
RLSC: Regularized Least-Squares Classification
SVM: Support Vector Machines
TBL: Transformation Based Learning
TCM: Tree Cut Model

TREC: Text Retrieval Conference
VSM: Vector Space Model
WSD: Word Sense Disambiguation
WSJ: Wall Street Journal

I. CHAPTER

Introduction

People like to control things and situations. They feel good when they are able to conquer new environments, when they become masters in worlds that were once arcane. The process of domination requires some effort and gives some rewards. For instance, little children like to hear the same story or watch the same video once and again until they totally learn it, and even after that, they feel happy getting back to it for a while. For adults similar things happen, we like to play sports at which we are good (and we get better doing it), and we like to know more details about domains in which we are experts (like arts or again, sports).

Both babies and adults feel rewarded when they start to talk and understand a language new to them. This feeling ends when the language is mastered, and then they do not think much of it. It becomes another ordinary controlled world; like walking, cooking, or playing cards. A new problem begins for us when we try to represent and model these conquered places. When we move to an environment that we have known for all our lifetime as (in my case) the Basque language, it seems that not much can escape to our control. We understand almost everything that it is said, and we can say whatever we feel inside. It looks like we would not have much trouble writing rules, or representations describing "how we do it". Some countries even have institutions devoted to prescribe how the language should be used, and for almost all languages there are listings of the meanings of the words and expressions that we use (dictionaries). It seems like we could try to write some code to mechanize the language understanding/producing process in a

computer.

However, we certainly cannot do that. Language can be seen as a discrete combinatorial system (Manning and Schütze, 1999), which means that from a fixed number of elements (words) combined, we can obtain infinite outcomes (sentences). Whatever path we take to solve the problem, the complexity arises immediately. These kind of problems, when humans try to go a step further and formalize the ways they interact with the worlds they control, are those faced by Artificial Intelligence (AI) research. Natural Language Processing (NLP) is one of those problems, and many researchers fall in love with this field precisely because "Natural Language" is the main tool we use to control one exciting world: the communication with other humans. The process that is happening at this moment when you go along these lines.

Thus, we can take any sentence as an example of communication between humans using Language as the code. If we could model the rules that lie behind the process, amazing applications could be created. Instead of a standard viewer, this text could be read with a Natural Language Understanding (NLU) tool[1]. The tool would understand the commands we give in English (in place of having to look for them in the menu), and also would understand the text that is written, allowing us to make queries like:

1. Can you get me a short version of this dissertation, of 20 pages more or less?

2. Where did the writer say something about parallel corpora?

3. Can you translate the whole document into Basque?

If you would ask these questions to a person that has read this dissertation carefully, he would understand the queries very easily, and, with more effort, he would have the ability to perform the requested tasks (assuming he knows English and Basque). For the hypothetical NLU tool, these tasks are still a long way ahead. As we said previously, this is a very appealing field for many researchers, and a big effort has been put in NLP. However, the more sophisticated applications that we can find in the market, such as Machine Translation (MT) Tools, Question Answering Systems, or Natural

[1]The NLU acronym is used in this book to refer to a software that would be able to perform processing, representation, and inference of the text that receives as input, imitating human capabilities. We will refer to NLP tools for programs that perform some of the intermediate tasks that would aid the hypothetical NLU tool.

Language Interfaces to Databases, do not show real understanding of human communication. For the NLU tool that would answer questions 1-3, the kind of reasoning required would be very complex to model. When processing the questions and the text of this book, the problem has been traditionally separated on these subtasks (Allen, 1995):

- Morphological analysis: how words are constructed from more basic meaning units called morphemes.
- Syntactic analysis: how words can be put together to form correct sentences. It determines what structural role each word plays in the sentence, and the phrase structure.
- Semantic analysis: the meaning of words and and how these meanings combine in the sentences.
- Pragmatic analysis: how sentences are used in different situations and how this affects the interpretation.
- Discourse analysis: how the immediately preceding sentences affect the interpretation of the next sentence.
- World knowledge: the general knowledge about the structure of the world that language users must have in order to fully understand the sentence.

Each of these subtasks receives a big number of researchers eager to test their approaches on them, using empirical approaches, or introspective rule-based methods. Each subtask can be divided into many others, for instance, for identifying phrases in a sentence, it would be important to know whether the words are verbs, nouns, or from other parts of speech (PoS). The hope is that these low-level tools constitute building blocks that will serve for the NLU applications of the future.

I.1 Word Sense Disambiguation (WSD)

I belong to a group (IXA[2]) interested in the processing of natural language, and that makes us get involved in many different aspects of this process.

[2]http://ixa.si.ehu.es

Personally, I find lexical ambiguity resolution a very interesting subtask of the big picture of NLU. This problem consists on determining the sense on which a word is used in a text, and in the NLP field is known as **Word Sense Disambiguation (WSD)**. As many other NLP tasks, this problem is not noticeable until you start to think on modeling it. People solves WSD constantly with many ambiguous words in each sentence. For instance, if we recall the first question we would give to the NLU tool "*Can you get me a short version of this dissertation, of 20 pages more or less?*", and focus on the word *pages*, every English speaker knows its meaning in the sentence. Moreover, if they were to look in a dictionary for more information, they would search for the lemma *page* as a noun. Below, we can see the list of meanings they would obtain from the WordNet lexical database (Fellbaum, 1998):

a. page – (especially one side of a leaf)

b. Page, Sri Frederick Handley Page – (English industrialist who pioneered in the design and manufacture of aircraft (1885-1962))

c. Page, Thomas Nelson Page – (United States diplomat and writer about the Old South (1853-1922))

d. page, pageboy – (a boy who is employed to run errands)

e. page – (a youthful attendant at official functions or ceremonies such as legislative functions and weddings)

f. page, varlet – (in medieval times a youth acting as a knight's attendant as the first stage in training for knighthood)

We know that the intended meaning in the sentence is the first on the list, but how could a program guess? There are many steps we have to make in order to process a text, but let us continue with the word *pages*, as it is. There are tools nowadays (lemmatizers or stemmers, and PoS taggers) that can tell us with good precision that *page* is the lemma of *pages*, and that it functions as a noun in the context of the question. This is not easy to do, as *pages* could be also a verb form; in order to do that, the PoS tagger has to know the contexts on which the word form appears, and solve the ambiguity. We will assume that we can use those tools (and others to come)

as building blocks in order to walk towards the tool this dissertation is about: an automatic WSD tool.

Following with the example, those nice lemmatizer and PoS taggers tell us that in question (1) we have a noun spelled *pages*, and that its root is *page*. Now, the NLU tool would need to know something more about this word in order to answer the question. What does *page* mean? Let us go back to the dictionary definitions. From the above list, we can discard b) and c), because they are in uppercase, suggesting that they are proper nouns, and *pages* is in lowercase; but how can we tell among the others?

As we will see throughout this dissertation, the problem is to build a model in a way that allows us to recognize the senses of the words in any of the infinite sentences that can be uttered. Focusing on the explicit disambiguation of word senses linked to a dictionary is not the only way to achieve understanding. Some authors describe the limitations of this fixed setting (Kilgarriff and Tugwell, 2002), and argue that a more dynamic approach (including a lexicographer in the loop) should be taken in order to represent the word meanings in a corpus. But whatever way we choose to obtain deep understanding of the text with automatic means, we think that the robust NLU interface should, in the end, be able to tell which one (or ones) from a given list of senses is (are) the closest to the intended meaning of *pages* in the context. In this dissertation, we will focus on the **explicit WSD approach, with a fixed list of senses**. We think that this line of research can provide fruitful insight into the deeper problem of NLU.

I.2 Approaches to WSD

Now that we have a fixed list of four noun senses to choose for the word *pages*, how do we approach the problem? One way to do it is to identify the senses the word can have in a dictionary or lexical resource (as we did for *page*), and construct a model of each sense. We can classify the WSD methods that follow this approach according to the knowledge they use to build the sense models:

- Based on hand-tagged corpora: the sentences where the different senses of the word are used.
- Machine Readable Dictionary (MRD) based: the information in the dictionary entries for each sense.

- Ontology based: the knowledge stored in an ontology, with its semantic relations with other senses.

- Other approaches: normally a combination of the previous sources, or other less explored sources (hand-built rules, for example).

Once we have information about the list of senses, the sentence that we are trying to understand is somehow compared to our sense-models. E.g. in order to know the sense of *pages* in the question "*Can you get me a short version of this dissertation, of 20* ***pages*** *more or less?*", we will compare the information we can extract from this sentence and contrast it with the data we have for each sense of *page* via corpora, MRDs, ontologies, or other means.

In order to represent the context of the occurrence we want to disambiguate, we extract *features* from the example. The features provide us the pieces of information that we will rely on to discriminate among the senses. We employ different tools to obtain them. For instance, some features that we could extract from question (1) for the target word *pages* are the following:

- Word-to-the-left *"20"* (Local feature)
- Lemma-bigram-in-context *"short version"* (Topical feature)
- *"page"* Head-of-PP-modifying *"version"* (Grammatical feature)
- Text type *scientific article* (Domain feature)

Some of the features will only require simple tools like tokenizers or stemmers to be extracted. Other tools that parse the sentences for grammatical dependencies, or that classify the text into domains are not as easy to obtain. As they are still object of research, we have to assume that some error will be introduced when we use these kind of features. However they will hopefully provide useful information about the context.

The selection of features is very important, as they have to reflect the relevant information in the contexts, and yet they have to be generic enough to be applied to a variety of cases. We will devote further attention to feature types in chapters III and IV.

As we said, state-of-the-art systems could be classified according to the knowledge source they use in order to learn their models. Another coarse distinction is usually applied between the systems that rely on hand-tagged

corpora (*supervised systems*), and those that do not require this resource (*unsupervised systems*). The former are called *supervised* because they need the supervision of a person that identifies the words in a sentence as pertaining to a sense or another. This distinction is important because the effort to tag the senses is high, and it would be costly to obtain tagged examples for all word senses and all languages, as some estimations show (Ng, 1997; Mihalcea and Chklovski, 2003). In this dissertation, **we will focus on supervised ML systems**. However, we will try to find alternatives to alleviate the hand-tagging cost in chapter VI.

I.3 State of the art in WSD

An important reference of the state of the art in WSD is the initiative for the Evaluation of Systems for the Semantic Analysis of Text, known as Senseval[3]. This competition held its third workshop on July, 2004 (the first edition took place in 1998, and the second in 2001). Senseval has been growing in languages, tasks, and participants over the years. We will dedicate more space to Senseval in chapter II. At this point we want to stress that regarding disambiguation performance, the results in the literature and in Senseval show that supervised ML is the most effective paradigm. Nevertheless, **current systems obtain around 70% accuracy** (Snyder and Palmer, 2004), which is not enough for practical applications. Some reasons that could explain these low scores are the following:

1. *The definition of the problem is wrong.* As we said, some authors claim that defining the meaning of a word as a discrete list of senses does not model correctly its behavior (Kilgarriff and Tugwell, 2002). There are suggestions that the instances of a word would be better represented as clusters when they have similar meanings, always in relation to a task or corpora (Kilgarriff, 1997). This issue has not been tackled here, and the classical discrete model has been adopted. This is the model followed by all of the current supervised WSD systems.

2. *Sense inventory and granularity.* In the last few years WordNet has been widely adopted as the sense-inventory of choice in the WSD community (Fellbaum, 1998), and WordNets for different languages have

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

been developed[4]. This resource has been applied in many of the Senseval tasks for English, and also for other languages like Basque, Italian, Spanish, etc. WordNet gives the possibility of comparing the results of different research groups, and offers a big conceptual network that can aid the disambiguation process, as well as manually tagged corpora. However, the sense inventory is clearly too fine-grained for many tasks and this makes the disambiguation very difficult. E.g. surely the 45 senses of *give* are not needed in a MT task, where not every sense will have a different translation for another language.

3. *ML algorithms are not adequately applied to the problem.* Methods coming from the ML community have been widely applied to the WSD problem: Naive Bayes, Decision Lists, AdaBoost, Support Vector Machines (SVM), etc. However, the comparative results show that even the most sophisticated methods have not been able to make a qualitative jump and get close to the solution of the problem. Actually, for each different word, different algorithms and features achieve the best results. Some voices claim that the optimization of ML methods, parameters, and feature types per word should help solve the problem (Hoste *et al.*, 2002).

4. *The feature sets used to model the language are too limited.* Traditionally simple feature sets consisting in bigrams, trigrams, and "bags of words" have been used to model the contexts of the target words. But in order to be robust, the ML methods should rely in as much information from the texts as possible. Features obtained with complex analysis of the text (morphological, syntactic, semantic, domain, etc.) and the combination of different types of features could be used.

5. *The sparse data problem.* In NLP most of the events occur rarely, even when large quantities of training data are available. This problem is specially noticeable in WSD, where hand-tagged data is difficult to obtain. Besides, fine-grained analysis of the context requires it to be represented with thousands of features, some of them very rare, but which can be very informative. Therefore the estimation of rare-occurring features is crucial to have high performance, and smoothing techniques can be useful in this process.

[4] http://www.globalwordnet.org/gwa/wordnet_table.htm

6. *Necessity of extra training data.* Existing hand-tagged corpora do not seem enough for current state-of-the-art systems. Hand-tagged data is difficult and costly to obtain. Estimations of the required tagging effort are not optimistic, and methods to obtain data automatically have not reached the same quality of hand-tagged data so far. Besides, the results reported usually in the literature are given for those words that have training examples, but a WSD tool should cover all the words in the vocabulary.

7. *Portability.* The porting of the WSD systems to be tested on a different corpora than the one used for training also presents difficulties. Previous work (Ng *et al.*, 1999; Escudero *et al.*, 2000c) has shown that there is a loss of performance when training on one corpora and testing on another. This has happened with automatically-tagged corpora, and also with corpora hand-tagged by independent teams of researchers. The problem could be alleviated using tuning methods, or taking into account the genre/domain of the corpora.

These issues represent a wide research space, and researchers from different fields have studied them from different perspectives. Our approach was to test empirically different ways to overcome some of the above problems:

(**1,2**) Regarding the first two points on the list on how will we define a word sense, we concentrated on the "discrete sense list" approach and chose the WordNet sense inventory whenever possible. As we said, WordNet has the advantage of being widely used in the community, and it offers important resources, as the Semcor all-words hand-tagged corpora (Miller *et al.*, 1993) and the conceptual hierarchy. Besides, this resource has been a meeting point for many research groups, by means of the Senseval settings, and also because of other collaborative projects: EuroWordNet (Vossen, 1998), Meaning (Atserias *et al.*, 2004), Germanet (Kunze and Lemnitzer, 2002), etc.

(**3**) The third issue on the list refers to the ML methods to apply. Early in this work, the state of the art in WSD showed that there was little difference in the performance of different ML algorithms. A method based on Decision Lists (DL) (Yarowsky, 1994) obtained the best performance in the Senseval-1 competition; this algorithm offered some advantages over other statistical methods like Naive Bayes (NB): as DLs are based on the best single evidence, in opposition to classification based on the combination of contextual

evidences, multiple non-independent features can be included in the system without having to model the dependencies. This factor would allow to explore feature-types coming from different analysis of the text (morphological, syntactic, semantic, etc.).

(**4**) We decided to apply the DL algorithm in order to focus on the 4th issue in the list (different types of features to model the context), instead of trying different ML methods. Our first hypothesis was that there was important information in the contexts of the words that was useful for learning, and that the integration of these features would contribute significantly to the resolution of the WSD problem.

As our work was going on, many approaches from the ML community were applied to WSD (AdaBoost, SVM, ...), and other methods, like the Vector Space Model (VSM), proved to be well suited for the task, performing significantly better than DLs. Thus, to be able to test our approaches with state-of-the-art methods, and compare our performance with other systems (cf. in the Senseval competitions), we incorporated some of these algorithms to our experiments.

(**5**) Regarding the sparse data problem in WSD, we also explored smoothing techniques to improve the estimation of the features in the training data using different ML methods. For our study, we were inspired by a method presented in (Yarowsky, 1995a). We implemented a method where the smoothed probabilities were obtained by grouping the observations by raw frequencies and feature types; and also by interpolation of the observed points.

(**6,7**) In order to make supervised WSD a realistic goal, our hypothesis was that the problem of the knowledge acquisition bottleneck (6th issue above) could be alleviated by automatic means. We analyzed the "monosemous relatives" method by Leacock *et al.* (1998), and tested it using the web as untagged corpus and the Senseval competition data for evaluation. We observed the difficulty of introducing new examples in a hand-tagged corpus, which took us to study the effect of the domain/genre of the corpora we use for learning and testing (7th problem in our list).

I.4 Contributions of the dissertation

Thus, our aim was to shed light on the needs of a WSD tool, and try to contribute to move the field forward. We explored two main hypotheses in this dissertation:

1. The use of richer features (syntactic, semantic, or domain features) can provide relevant information of the contexts, and it should improve significantly baseline methods that are trained on classic features.

2. The automatic acquisition of examples by means of WordNet relatives can alleviate the knowledge acquisition bottleneck, and improve over other unsupervised (or minimally supervised) approaches.

We proposed different ways to explore these issues, developing approaches not previously described in the literature. All in all, we think that our main contributions on these initial hypotheses are the following:

- **Syntactic features (chapter IV)**: We explored the contribution of an extensive set of syntactic features to WSD performance. We presented several experiments and analyses on these features. The study included two different ML methods (DL and AdaBoost (AB)), and a precision/coverage trade-off system using these feature types. The results show that basic and syntactic features contain complementary information, and that they are useful for WSD. The contribution of this type of features is specially noticeable for the AB algorithm in the standard setting, and for DLs when applying the precision/coverage trade-off.

- **Semantic features (chapter IV)**: We applied two approaches to study the contribution of semantic features using the WordNet hierarchy and the Semcor all-words corpus. On the one hand, we constructed new feature types based on the synsets surrounding the target word, the hypernyms of these synsets (at different levels), and also their semantic files. On the other hand, we learned different models of selectional preferences for verbs, using the relations extracted from the Semcor corpus by Minipar. Our main conclusions were that the "bag-of-synsets" approach does not seem to benefit much from the WordNet hierarchy. Instead, selectional preference acquisition offers promising results with a view to their integration with other feature types.

- **Automatic acquisition of examples (chapter VI)**: We evaluated up to which point we can automatically acquire examples for word senses and train supervised WSD systems on them. The method we applied is based on the monosemous relatives of the target words (Leacock *et al.*, 1998), and we studied some parameters that affect the quality of the acquired corpus, such as the distribution of the number of training instances per each word sense (bias). We built three systems with different supervision requirements: fully supervised (automatic examples added to hand-tagged corpora), minimally supervised (requiring information about sense distributions), and unsupervised (without hand-tagged examples). We showed that the fully supervised system combining our web corpus with the examples in Semcor improves over the same system trained on Semcor alone (specially for nouns with few examples in Semcor). Regarding the minimally supervised and fully unsupervised systems, we demonstrated that they perform well better than the other systems of the same category presented in the Senseval-2 lexical-sample competition. Our system can be trained for all nouns in WordNet, using the data collected from the web.

- **Genre/topic shift (chapter VII)**: We studied the strength of the "one sense per collocation" hypothesis (Yarowsky, 1993) using different corpora for training and testing. Our goal was to measure the importance of introducing examples from different sources in WSD performance. We focused on the domain/genre factor, and performed our experiments in the DSO corpus, which comprises sentences extracted from two different corpora: the balanced BC, and the WSJ corpus containing press articles. Our experiments show that the one sense per collocation hypothesis is weaker for fine-grained word sense distinctions, and that it does hold across corpora, but that collocations vary from one corpus to other, following genre and topic variations. This would explain the low performance for WSD across corpora. In fact, we showed that when two independent corpora share a related genre/topic, the WSD results are better.

Other interesting results that came out from our work on this dissertation are the following:

- **High-precision WSD tool for English (chapter IV)**: We tested on Senseval-2 data different systems that could provide high precision

at the cost of coverage. The results were promising, as two methods based on DLs reached 93% precision at 7% coverage (decision-threshold method), and 86% precision at 26% coverage (feature selection method). Syntactic features are specially helpful for feature selection.

- **Supervised WSD tool for English (chapter V)**: We developed a supervised system based on the combination of different ML methods and in smoothing techniques, described in chapter V. In the Senseval-3 English lexical-sample task, it ranked 5th among 47 submissions, only 0.6% lower than the best system. This system also participated in the all-words task, as a component of the "Meaning" system, which ranked 5th among 26 systems.

- **Supervised WSD tool for Basque (chapter V)**: We have adapted our models to Basque, which is an agglutinative language and presents new challenges when defining the feature set. We have tested this tool on the Senseval-3 Basque lexical-sample task data, and it outperforms the results of other systems that took part in the event.

- **Unsupervised WSD tool for English (chapter VI)**: We built an unsupervised system relying on automatically obtained examples, which shows promising results for alleviating the knowledge acquisition bottleneck. It has been tested on the Senseval-2 English lexical-sample task, presenting the best performance among systems of this kind.

There are also some resources (available for research) that have been developed as a result of our work:

- **Selectional preferences (chapter IV)**: Using the syntactic dependencies (object and subject) extracted from Semcor, we constructed and evaluated selectional preferences for verb and noun classes in WordNet. This database, consisting on weighted relations between synsets, is available by means of a Meaning license, or by personal request.

- **Sense tagged corpus (chapter VI)**: We constructed automatically a sense-tagged corpus for all nouns in WordNet. This resource is publicly available, and can be downloaded from
`http://ixa2.si.ehu.es/pub/sensecorpus`.

Finally, during this research, we have published our results in different articles. The complete list is given in appendix A.

I.5 Structure of the dissertation

In summary, this is the structure of this dissertation, and the issues addressed by each chapter:

- **First chapter.** This introduction.
- **Second chapter.** State of the art: resources, systems, and evaluation. This chapter will be devoted to the description of different methods and research lines that are presenting promising results in the WSD task. First, we will describe the main resources that are employed for WSD, including lexical databases, corpora, and some well-known learning algorithms. The main sections will be dedicated to the Senseval competitions and the participating systems.
- **Third chapter.** Baseline WSD system: DL and basic features. In this chapter, we will study the DL algorithm, trained on "classic" feature types and currently available hand-tagged data; all in an extensive set of experiments. The tests and results will serve as a reference for the following chapters, which focus on different aspects of the disambiguation task.
- **Fourth chapter.** New feature types: syntactic and semantic knowledge. In this chapter we will analyze richer feature sets: syntactic features, semantic features, and selectional preferences. We will study the contribution of different feature types to the disambiguation process, relying on two different ML methods. We will also explore the contribution of new features to a high precision WSD system.
- **Fifth chapter.** Sparse data problem and smoothing techniques. Different smoothing techniques will be applied to a set of ML algorithms. The goal of this chapter will be to obtain better estimations of features for improved WSD, helping to alleviate the sparse data problem. We will also analyze the behavior of different ML methods and their combination in order to improve performance.

- **Sixth chapter.** Automatic acquisition of sense-tagged examples. Application of the "monosemous relatives" method (Leacock *et al.*, 1998) for automatically acquiring sense-tagged examples. These examples, and an automatic ranking of senses obtained with the method by McCarthy *et al.* (2004) will be used to build different systems to be evaluated on the Senseval-2 dataset.

- **Seventh chapter.** Portability and genre/topic of corpora. The goal of this chapter was to measure the importance of introducing examples from different sources in WSD performance. We focused on the "one sense per collocation" hypothesis, and the effect of the genre and topic of the training and testing datasets.

- **Eighth chapter.** Conclusions and future work. This last chapter summarizes the main conclusions of the dissertation and sketches the further work on the opened research lines.

II. CHAPTER

State of the art: resources, systems, and evaluation

In this chapter we will present the state of the art for WSD. As we will see, this is a task that has received a great deal of attention from many researchers in NLP during the years. Because an extensive survey of all these works is out of the scope of this dissertation, we will organize the chapter as follows. First we will briefly introduce previous work on WSD, and also justify our organization of the analysis of the literature. The next section will describe the main resources that are applied to WSD research: on the one hand, lexical databases and dictionaries that are used as sense repository; on the other hand, publicly available corpora that is employed by the systems for learning. The next section will be devoted to present some well-known algorithms that have been applied to WSD, and which will be employed in different experiments throughout this dissertation. After that, we will present measures and significance tests that are used to evaluate WSD systems. The final three sections of this chapter will be dedicated to the Senseval competitions and the participating systems, focusing on the English tasks.

II.1 Introduction

There has been a vast corpus of work on WSD since the fifties. The history of NLP is very much linked to this task, which was first treated independently

in 1955 (Yngve, 1955). This happened in the days of the big investment for MT in the United States of America. The lexical ambiguity problem arose immediately, as it happens when we try to construct applications that require deep understanding of the texts. In fact, evident errors appear when this problem is not addressed in some way, and the folklore of early AI keeps stories like the biblical English sentence *The spirit is strong, but the flesh is weak* being translated from English into Russian, and back into English. According to the story, the resulting sentence turned out to be *The vodka is good, but the meat is rotten.* This can seem a bit exaggerated, but even nowadays, if we use a commercial MT tool, these kind of errors will appear[1]

The notorious complexity of lexical disambiguation was one of the arguments that were raised against NLP funding in the early days of AI. One famous example came from Bar-Hillel's work (Bar-Hillel, 1960), who claimed that it was impossible for a machine to disambiguate correctly between two main senses of the word *pen* (the writing device, or the enclosed space) in the sentence *"the box is in the pen"*. A few years after that, the ALPAC report was released (ALPAC, 1966), exposing the weakness of the work on MT, and stopping funding for NLP. The work on NLP was then re-oriented to knowledge representation and semantic networks, and there is where WSD had its space until the eighties, when the availability of large corpora and the advent of Internet changed the trend to empirical approaches again.

In recent years, extensive literature on WSD has been developed. Even if WSD is not a final NLP application, but an intermediate task such as PoS tagging or parsing, there are some factors that make it attractive to researchers:

- The problem can be modeled as another classification task, and approaches from the ML community can be applied.
- It is easy to evaluate new systems against existing goldstandards.
- The work with word senses is very much linked to the cognitive process of representing concepts and knowledge.
- Solving the lexical disambiguation problem should have an immediate effect in NLP tools, and create new ones.

[1]For instance, if we use the on-line demo of Systran 5.0 (http://www.systransoft.com) to translate the example into Spanish, and back into English, we will come out with "*The alcohol is strong, but the meat is weak*". We tried Russian first, and the result was the cryptic sentence "*The spirit of sil'n, but the flesh is weak.*".

- The impulse that the WordNet resource and the Senseval competitions are giving to this task, joining the efforts of many groups of researchers in this area.

Thus, general NLP books dedicate separate chapters to WSD (Manning and Schütze, 1999; Jurafsky and Martin, 2000; Dale *et al.*, 2000). There are also special issues on WSD in NLP journals (Ide and Veronis, 1998; Edmonds and Kilgarriff, 2002); and books devoted specifically to this issue (Ravin and Leacock, 2001; Stevenson, 2003; Agirre and Edmonds, forthcoming). Traditionally, as we have seen in chapter I, WSD systems are classified attending to the type of knowledge they rely on to build their models (hand-tagged corpora, raw corpora, MRDs, ontologies, or combination of those sources). Another distinction that is usually made, for instance in the Senseval competitions, is between supervised and unsupervised methods. However, nowadays it is difficult to present a strict classification. For example, in Senseval we can see systems that do not learn directly from the examples, but use their prior distribution; also systems that use only a minimal number of examples to link the classes they induce to the sense inventory; and even semi-automatic architectures that rely on lexicographers providing clues for disambiguation by hand.

As this field covers a huge space, the development of an exhaustive survey of WSD is out of the scope of this dissertation, and we have to focus on the aspects of the problem that interest us most, those that have been listed on the introduction chapter. We will pay special attention to the most successful systems for WSD, both in the literature and in the Senseval competitions: supervised ML techniques. However, other promising approaches for WSD will be presented together with the corresponding motivations in relation to our work. For a full account of the field please turn to the above references.

The Senseval workshops are the best reference to see where the field is moving, and they will serve us to organize this chapter. At the moment of writing this dissertation, the third edition in Barcelona was just finished. Fruitful discussion resulted from the analysis of the results in the workshop, with leading researchers on the field sharing their views[2]. We will present the state-of-the-art in WSD following the path of the three Senseval competitions celebrated until this day (1998, 2001, and 2004), focusing on the supervised systems in the English lexical-sample and all-words tasks.

[2]The following mailing list (Senseval-discuss) provides additional information: http://listserv.hum.gu.se/mailman/listinfo/senseval-discuss.

As we will see, the best results in the last Senseval were achieved by systems that rely in different ML techniques (kernel-based, optimization, and voting), and in most cases include rich information from the context, like dependency relations or domain information. After we present the basic resources, evaluation measures, and single algorithms, we will describe the top-performing systems for English in the different Senseval editions. Other approaches and tasks that are related in some manner to our work (from Senseval or not) will be described in the "related work" sections of the different chapters. We will briefly mention here the works covered in other chapters:

- **Fourth chapter:** This chapter, involving the baseline setting of the dissertation, will introduce experiments by other research groups on the following subjects: local vs topical features, learning curves, and performance on an all-words corpus.

- **Fifth chapter:** This chapter is devoted to WSD literature that relies on a context representation model that goes beyond the classic set of features. Works that use syntactic dependencies, selectional preferences, or domain information will be analyzed.

- **Sixth chapter:** An inherent problem of WSD is the lack of tagged examples per sense. This chapter revises work that try to obtain the most of sparse features by means of different techniques to smooth probabilities. Architectures that use combinations of single algorithms by voting are also presented here.

- **Seventh chapter:** In this chapter, research on ways to alleviate the knowledge acquisition bottleneck will be presented. Works on automatic acquisition of tagged examples, active learning, and bootstrapping are included in this section. As this line of investigation aims at using the minimal human supervision, other unsupervised works with successful performance in the Senseval competition are introduced.

- **Eighth chapter:** This chapter is dedicated to studies on training and testing in different corpora. Here we will see tuning methods to overcome the usual decrease in performance, and also ways to adapt the sense inventory to the domain of the text.

II.2 Lexical databases and dictionaries

In this section we will introduce the main lexical repositories that have been used in the Senseval editions to provide the sense inventories for English and Basque. These resources have been used for our experiments throughout the dissertation.

WordNet lexical database

WordNet (Fellbaum, 1998) is a lexical database developed at Princeton University[3]. This semantic network is connected with paradigmatic relations, such as synonymy, hyperonymy, antonymy, and entailment. All English open-class words are included in this resource. The concepts are represented by *synsets*, which store the words that are synonymous in some context, e.g. {*bank, cant, camber* }[4] .The main relation that structures this database is hyperonymy, which gives a hierarchical organization to WordNet for verbs and nouns (adjectives and adverbs are organized differently).

WordNet is widely used in NLP research, specially for WSD. The sense distinctions in WordNet have become a commonplace for WSD research since they were adopted in the Senseval-2 competition; although the sense inventory has been criticized for its fine-grainedness, specially for verbs.

There have been different versions of WordNet during the years, and mappings between versions (Daude *et al.*, 2000) have been developed in order to use different resources (such as hand-tagged corpora and WordNets in other languages). The current version (August, 2004) is 2.0 . Table II.1 shows the corpora used for WSD that have been tagged with different WordNet versions. These corpora will be described in detail in section II.3.

As we mentioned in the introduction, WordNets for different languages have been developed and linked to the original Princeton WordNet. Many languages have adopted the WordNet sense inventory to organize Senseval tasks, and therefore hand-tagged data has been built for other languages, keeping the connection to English. The linking of WordNets offers interesting prospects, making possible to experiment with multilingual information, as different projects have shown (Atserias *et al.*, 2004; Vossen, 1998). The

[3]The original WordNet is sometimes referred as "Princeton WordNet", to distinguish it from other extensions of this approach.

[4]The synsets are usually represented by the word list between brackets.

Corpus	WordNet version
DSO	1.5 (Pre)
Semcor	1.6
Senseval-2 all-words	1.7 (Pre)
Senseval-2 lexical-sample	1.7 (Pre)
Senseval-3 all-words	1.7.1
Senseval-3 lexical-sample (except verbs)	1.7.1

Table II.1: Publicly available hand-tagged corpora and WordNet versions for English. (Pre) indicates that a preliminary version of WordNet was utilized at the moment of tagging.

Basque WordNet[5] is one of the resources being built and connected to Princeton WordNet (version 1.6) . This resource was used as sense inventory for the Basque lexical-sample task in Senseval-3 (cf. section II.3). Some of the words chosen for Basque were translations of the words in the English lexical-sample task.

HECTOR lexical database

HECTOR (Atkins, 1993) is a research project for the development of a database linked to a dictionary and a hand-tagged corpus. The dictionary entries were built by lexicographers in a corpus-driven approach. The results for a sample of words were used in the first Senseval edition. A pilot of the British National Corpus[6] (BNC), comprising 17 million words was chosen to retrieve the examples to tag.

Euskal Hiztegia *Basque dictionary*

Euskal Hiztegia (Sarasola, 1996) was chosen for the first edition of the Senseval Basque task (which was held in Senseval-2). At the time, this monolingual dictionary was the one available in MRD form for Basque. The dictionary has 30,715 entries and 41,699 main senses.

[5]The Basque WordNet is available at http://ixa3.si.ehu.es/wei3.html
[6]http://www.natcorp.ox.ac.uk

II.3 Corpora

This section is devoted to the main sources of hand-tagged corpora that are used to build supervised WSD systems. We will introduce widely-used resources that are available for research. For many years, these resources were limited to a few projects for English (see Semcor and DSO below). However, in recent years the Senseval initiative has made a qualitative jump, providing hand-tagged data for different languages and tasks.

Semcor corpus

Semcor (Miller *et al.*, 1993) consists on a subset of the Brown Corpus (BC) plus the novel *The Red Badge of Courage*. It contains a number of texts comprising about 200,000 words where all content words have been manually tagged with senses from WordNet 1.6 . It has been produced by the same team that created WordNet. Semcor has been cited as having scarce data to train supervised learning algorithms (Miller *et al.*, 1994). More details on this corpus can be found in the experiments performed in chapters III and IV, or in (Francis and Kucera, 1982).

DSO corpus

The Defense Science Organization (DSO) corpus was differently designed (Ng and Lee, 1996). 191 polysemous words (nouns and verbs) of high frequency were selected from the Wall Street Journal (WSJ) and Brown Corpus (BC). A total of 192,800 occurrences of these words were tagged with WordNet 1.5 senses[7], more than 1,000 instances per word in average. The examples from the BC comprise 78,080 occurrences of word senses, and the examples from the WSJ consist on 114,794 occurrences.

It is important to note that the BC is balanced, and the texts are classified according to some predefined categories (the complete list is shown in table II.2). The BC manual (Francis and Kucera, 1964) does not detail the criteria followed to set the categories:

> *The samples represent a wide range of styles and varieties of prose... The list of main categories and their subdivisions was*

[7] A previous version of WordNet 1.5 was used at the moment of tagging, and there are slight differences with the final 1.5 version.

Informative prose
A. Press: Reportage
B. Press: Editorial
C. Press: Reviews (theater, books, music, dance)
D. Religion
E. Skills and Hobbies
F. Popular Lore
G. Belles Lettres, Biography, Memoirs, etc.
H. Miscellaneous
I. Learned
Imaginative prose
J. General Fiction
K. Mystery and Detective Fiction
L. Science Fiction
M. Adventure and Western Fiction
N. Romance and Love Story
O. Humor

Table II.2: List of categories of texts from the Brown Corpus, divided into informative prose (top) and imaginative prose (bottom).

drawn up at a conference held at Brown University in February 1963.

Regarding the WSJ corpus, all the texts come from press articles. More details on DSO can be found in the experiments in chapters III, IV, and VII.

Senseval-1 English lexical-sample corpus

This corpus (Kilgarriff and Rosenzweig, 2000) consists on 8,512 test instances and 13,276 training instances for 35 words (nouns, verbs, and adjectives). The instances are tagged with HECTOR senses (cf. section II.2), and their polisemy ranges from 2 to 15 senses. The examples are extracted from a pilot of the BNC. The list of words and the number of testing examples per word can be seen in the appendix (cf. table B.1).

Senseval-2 English lexical-sample corpus

This corpus (Kilgarriff, 2001) consists on 73 target words (nouns, verbs, and adjectives), with 4,328 testing instances, and 8,611 training instances. The

examples come from the BNC (mostly), and from the WSJ. The chosen sense inventory was a previous version of WordNet 1.7 (1.7 pre-release), specially distributed for this competition. The complete list of words is given in the appendix (cf. table B.2). A peculiarity of this hand-tagged corpus is that the examples for a given target word include multiword senses, phrasal verbs, and proper nouns. In order to process these cases, we can include them as regular sense–tagged examples, we can remove them, or we can try to detect them by pre-processing (cf. section III.3.2).

Senseval-2 English all-words corpus

The test data for this task (Palmer *et al.*, 2001) consists on 5,000 words of text from three WSJ articles representing different domains from the Penn Tree-Bank II. The sense inventory used for tagging is the WordNet 1.7 pre-release. All content words are sense-tagged, including multi-word constructions. Experiments on this corpora are described in chapter III.

Senseval-2 Basque lexical-sample corpus

The complete corpus (Agirre *et al.*, 2001) consists on 5,284 hand-tagged occurrences of 40 words (nouns, verbs, and adjectives), from which 2/3 were separated for training and the rest for evaluation. The sense inventory was obtained from Euskal Hiztegia (cf. section II.2). The instances have been extracted from two corpora: a balanced corpus, and articles from the newspaper *Egunkaria*. Some instances are tagged with multiwords. The list of words, with their frequency and polisemy degree, can be seen in the appendix (cf. table B.3). Experiments on this data are described in chapter III.

Senseval-3 English lexical-sample corpus

This corpus (Mihalcea *et al.*, 2004) was built relying on the Open Mind Word Expert system (Mihalcea and Chklovski, 2003). Sense tagged examples were collected from web users by means of this application. The source corpora was BNC, although early versions included data from the Penn TreeBank corpus, the *Los Angeles Times collection*, and *Open Mind Common Sense*. As sense inventory WordNet 1.7.1. was chosen for nouns and adjectives, and

the dictionary *Wordsmyth*[8] for verbs. The main reason to rely in another inventory for verbs was the fine-grainedness of WordNet. The results for verbs are usually poor, and they wanted to test the effect of using a coarser inventory.

57 words (nouns, verbs, and adjectives) were tagged in 7,860 instances for training and 3,944 for testing (see list in the appendix, table B.4). Experiments on this corpus are described in chapter V.

Senseval-3 English all-words corpus

As in Senseval-2, the test data for this task consisted on 5,000 words of text (Snyder and Palmer, 2004). The data was extracted from two WSJ articles and one excerpt from the BC. The texts represent three different domains: editorial, news story, and fiction. Overall, 2,212 words were tagged with WordNet 1.7.1. senses (2,081 if we do not include multiwords).

Senseval-3 Basque lexical-sample corpus

For this corpus (Agirre *et al.*, 2004), 40 words (nouns, verbs, and adjectives) were tagged in 7,362 instances (2/3 were distributed for training, the rest for evaluation). The chosen sense inventory was the Basque WordNet, which is linked to the version 1.6 of Princeton WordNet (cf. section II.2). Examples tagged with multiword senses were included. Together with this data, they also distribute 62,498 untagged examples of the 40 words, obtained from the Internet.

The hand-tagged corpora is extracted from three sources: a balanced corpus, the newspaper *Egunkaria*, and the Internet. The corpus distribution includes linguistic processing, such as lemmatization, PoS tagging, and identification of case markers (Basque is an agglutinative language). Instances that are tagged with multiwords are kept. The complete list of words and their frequencies can be seen in the appendix (cf. table B.5). Experiments on this corpus are described in chapter V.

[8] http://www.wordsmyth.net/

II.4 Learning algorithms

We will present here different methods that are used widely for supervised WSD, alone or in combination. Most of our experiments are performed using the first algorithm (Decision Lists), but the other methods will also be applied in different parts of this dissertation.

As we mentioned in section I.2, in order to represent the context of the word occurrence we want to disambiguate, we extract features (f) from the example using different tools. Then, the ML methods below return a weight for each sense ($weight(s_k)$), and the sense with maximum weight is selected.

II.4.1 Most Frequent Sense baseline (MFS)

This simple baseline method is frequently applied in WSD literature. It consists on counting the number of examples for each sense in training data, and assigning the most frequent to all the examples in testing. In case of ties, the algorithm chooses at random. Despite its simplicity, this approach is difficult to beat for all-words systems that do not rely on hand-tagged data.

II.4.2 Decision Lists (DL)

A Decision List consists of a set of ordered rules of the form (feature-value, sense, weight). In this setting, the Decision Lists algorithm works as follows: the training data is used to estimate the features, which are weighted with a log-likelihood measure (Yarowsky, 1995b) indicating the likelihood of a particular sense given a particular feature value. The list of all rules is sorted by decreasing values of this weight. When testing new examples, the decision list is checked, and the feature with highest weight that matches the test example selects the winning word sense.

The original formula (Yarowsky, 1995b) can be adapted in order to handle classification problems with more tan two classes. In this case, the weight of sense s_k when feature f occurs in the context is computed as the logarithm of the probability of sense s_k given feature f divided by the sum of the probabilities of the other senses given feature f. That is, the weight of s_k is obtained by the following formula:

$$weight(s_k) = \arg\max_f \ \log\left(\frac{P(s_k|f)}{\sum_{j\neq k} P(s_j|f)}\right) \qquad \text{(II.1)}$$

These probabilities can be estimated using the maximum likelihood estimate, and some kind of smoothing so as to avoid the problem of 0 counts. Different approaches for smoothing have been explored in chapter V. As default, a very simple solution has been adopted, which consists of replacing the denominator by 0.1 when the frequency is zero. This value was determined empirically in previous experiments.

In some cases, for a given feature, there is only one occurrence, or the weight for all the senses is lower than zero. Another decision if whether to include these features in the decision list or not. We call this parameter *pruning*, and in most of the experiments we will apply *pruning*, that is, we will discard these features. In the experiments where we need higher coverage we will not use *pruning*, and we will indicate it explicitly.

II.4.3 Naive Bayes (NB)

The Naive Bayes (NB) method is based on the conditional probability of each sense s_k given the features f_i in the context. It assumes independence of the features, which is not real, but it has been shown to perform well in diverse settings (Mooney, 1996; Ng, 1997; Leacock *et al.*, 1998). The sense s_k that maximizes the probability in formula II.2 is returned by the algorithm.

$$weight(s_k) = P(s_k) \ \prod_{i=1}^{m} P(f_i|s_k) \qquad \text{(II.2)}$$

The values $P(s_k)$ and $P(f_i|s_k)$ are estimated from training data, using relative frequencies. It requires smoothing in order to prevent the formula from returning zero because of a single feature. A method that has been used in some previous work with this algorithm (Ng, 1997; Escudero *et al.*, 2000b) is to replace zero counts with $P(s_k)/N$, where N is the number of examples in training. We used this method as default smoothing. This algorithm has been applied in section IV.9 with semantic features, and in the experiments on smoothing in chapter V.

II.4.4 Vector Space Model (VSM)

For the Vector Space Model (VSM) method, we represent each occurrence context as a vector, where each feature will have a 1 or 0 value to indicate the occurrence/absence of the feature. For each sense in training, one centroid vector is obtained ($\vec{C_{s_k}}$). These centroids are compared with the vectors that

represent testing examples ($\vec{f}$), by means of the cosine similarity function (formula II.3). The closest centroid assigns its sense to the testing example. No smoothing is required to apply this algorithm, but it is possible to use smoothed values instead of 1s and 0s, as we will see in chapter V.

$$weight(s_k) = cos(\vec{C_{s_k}}, \vec{f}) = \frac{\vec{C_{s_k}} \cdot \vec{f}}{|\vec{C_{s_k}}||\vec{f}|} \quad \text{(II.3)}$$

II.4.5 Support Vector Machines (SVM)

Regarding Support Vector Machines (SVM), we utilized SVM-Light, a public distribution of SVM by Joachims (1999), in order to test the SVM method (Vapnik, 1995) in our setting. The basic idea of the algorithm is to use the training data to learn a linear hyperplane that separates the positive examples from the negative examples. The location of this hyperplane in the space is the point where the distance to the closest positive and negative examples (the margin) is maximum. In some cases, it is not possible to obtain an hyperplane that divides the space linearly, or it is worthy to allow some errors in training data to construct a more efficient hyperplane. This can be achieved with the "soft margin" variant of the method, which permits a trade-off between training errors and the maximization of the margin. The "soft margin" variant requires the estimation of a parameter (denoted as C). We estimated the C using a greedy process in cross-validation on the training data. The weight for each sense is given by the distance to the hyperplane that supports the classes, that is, the sense s_k versus the rest of senses ("one vs all" approach). This method has been applied in chapter V.

II.4.6 AdaBoost (AB)

AdaBoost (AB) is a general method for obtaining a highly accurate classification rule by linearly combining many weak classifiers, each of which may be only moderately accurate (Freund and Schapire, 1997). For our experiments, a generalized version of the AB algorithm has been used, (Schapire and Singer, 1999), which works with very simple domain partitioning weak hypotheses (decision stumps) with confidence rated predictions. This particular boosting algorithm is able to work efficiently in very high dimensional feature spaces, and has been applied, with significant success, to a number of NLP disambiguation tasks, including WSD (Escudero *et al.*, 2000a).

Regarding parametrization, the smoothing parameter has been set to the default value (Schapire and Singer, 1999), and AB has been run for a fixed number of rounds (200) for each word. No optimization of these parameters has been done at word level. When testing, the sense with the highest prediction is assigned. This method has been applied with syntactic features in chapter IV.

II.5 Evaluation

In order to evaluate how well do the systems perform, hand-tagged corpora is used as gold standard, and different measures are calculated comparing the answers of the system to this gold standard. Depending on the corpora we use, two approaches have been taken for evaluation.

- One training/test partition: one part of the corpus is used for learning, and the rest for evaluation. This approach is applied with the Senseval datasets, and in cross-corpora tagging experiments.
- Cross-validation: the corpora is split in N parts of similar size, and this process is repeated for each of the pieces in turn: the chosen part is used as gold-standard, and the remaining (N-1) parts for training the system. The final result is the average of the N executions. We can partition the corpora randomly, or in a stratified way, that is, trying to keep the same proportion of word senses in each of the folds. Cross-validation is used when working on Semcor or DSO.

II.5.1 Measures

In order to measure the goodness of WSD methods, we use the following measures: precision, recall, coverage, and F1 (harmonic average between precision and recall), all ranging from 0 to 1. Given N (number of test instances), A (number of instances which have been tagged), and C (number of instances which have been correctly tagged):

- $precision = C/A$
- $recall = C/N$
- $coverage = A/N$

- $F1 = (2 * precision * recall)/(precision + recall) = (2 * C)/(A + N)$

When multiple senses are chosen, we use a modified measure of precision, equivalent to choosing at random in ties. Instead of counting 1 when any of the winning senses is correct, we count only a fraction. That is, we substitute C with C' in the above formula, where C' is computed as follows:

$$C' = \sum_{i \in test\ instances} C(i) \quad \text{where} \quad C(i) = \begin{cases} 1 & \text{if instance i correct} \\ 0 & \text{otherwise} \end{cases}$$

The results will be given in percentage points. Except for the initial experiments, the results will be rounded to the first decimal number, to be able to differentiate when differences are small. In the initial experiments only integer values are provided.

In the Senseval competition a similar evaluation schema is applied. There are slight differences when there are multiple correct labels and multiple answers. The Senseval scoring software incorporates some ideas from (Resnik and Yarowsky, 1997).

Finally, in most of the tables, the results averaged by PoS or overall are shown. In order to obtain these values, the number of examples for each word is always used to micro-average the results. For example, if we have only two adjectives in a word-set (e.g. *all* (200 occurrences, 90% precision) and *long* (100 occurrences, 60% precision)), in order to obtain the average precision for adjectives, we proceed as follows:

$$avg.\ precision\ for\ adjs. = \frac{prec.(all) * freq(all) + prec.(long) * freq(long)}{freq(all) + freq(long)}$$

$$= \frac{90\ *\ 200\ +\ 60\ *\ 100}{200\ +\ 100} = 80.0$$

II.5.2 Significance tests

When comparing the performance of two algorithms, there are statistical tests that help us to know whether the precision or recall difference we observe is significant. A comparison of these methods can be found in (Dietterich, 1998). We will apply two of these tests in some of our experiments:

- McNemar's test: it is employed when there is only one execution of the algorithm, that is, when the training and testing data do not change. The method is based on a χ^2 test for goodness-of-fit that compares the distribution of correct/incorrect counts expected under the null hypothesis (same error rate for both algorithms) to the observed counts.

- Cross-validated paired Student t test: it is applied when the evaluation is based on cross-validation. In this case, the null hypothesis is that the difference in the error of each classifier for each partition is drawn independently from a normal distribution. The Student's t statistic gives us the value of the actual distribution and the threshold to reject/accept the null hypothesis.

II.6 WSD systems in Senseval-1

The first edition of Senseval took place in 1998 at Herstmonceux Castle, England (Kilgarriff, 1998). For the first time, the research community made a joint effort to define the procedure and methodology for the evaluation of such a controversial task as WSD. The goal was to follow the example of other successful competitive evaluations, like MUC (Message Understanding Conference) or TREC (Text Retrieval Conference). In this first edition, extensive discussion was carried out on issues as the lexicon, the tagging methodology, the evaluation procedure, or even the appropriateness of defining WSD in the way it was. Finally, three tasks were arranged that consisted in tagging a predefined set of words (lexical-sample) for three languages: English, French, and Italian. A total of 25 systems from 23 research groups made it to the final schedule.

The systems that competed in Senseval-1 relied on ML methods (using the available training data to extract features), or in lexical resources such as WordNet. Prior to the event, there were two types of systems being developed for WSD (supervised and unsupervised), and it was not clear which would achieve better performance. During the competition, many combined approaches were presented, relying both in hand-tagged corpora and other lexical resources, looking for robustness rather than "purity". However, clearly the top-scoring systems were those relying on hand-tagged data for training their models.

Focusing on the English lexical-sample task, which had the highest par-

ticipation, the results of the best systems ranged between 74%-78% recall for fine-grained scoring, while a baseline method based on (Lesk, 1986) achieved 69% recall. We will describe briefly this baseline method, and the three top-scoring systems for this task[9].

Lesk-corpus (Lesk, 1986)

Method based on the overlapping between the target context, and the definitions plus the tagged examples for each sense. Although simple, the use of training examples to construct the model makes it difficult to beat, specially for unsupervised approaches.

JHU (Yarowsky, 2000)

This system, which had the best score after re-submission with 78.1% recall, was a supervised algorithm based on hierarchies of DLs. The method tries to take advantage of the conditional branching at the top levels of the Decision Tree approach, while avoiding the data fragmentation problem. It relies on a rich set of features (collocational, morphological, and syntactic) to classify the examples. It also associates weights to the different types of features.

Durham (Hawkins and Nettleton, 2000)

The system called "Durham" was the best scoring after the first submission of systems, attaining 77.1% recall. It consisted on a hybrid approach relying on three types of knowledge: stochastic (frequency of senses in training data), rule-based (clue words from the training context), and sub-symbolic (contextual similarity between concepts). One of the main drawbacks of this system was the requirement of hand-work in order to obtain clue words from the context of the target words.

Tilburg (Veenstra et al.*, 2000)*

This team applied a Memory Based Learning (MBL) method to retrieve the closest match to the test example from the training instances; then the sense of the training instance is assigned. This method achieved 75.1% recall.

[9]There was the option to resubmitting the results after correcting bugs, and therefore there were 2 official scores. In any case, the three best systems remained the same.

This ML approach does not require to perform a generalization step with the hand-tagged data. In this case, automatic feature-weighting was applied in the similarity metric, and a word expert was built for each target word in Senseval. The word experts were constructed by exhaustive search on training data by 10 fold cross-validation, attending to these factors:

- Variant of the learning algorithm
- Parameter setting
- Feature construction setting

II.7 WSD systems in Senseval-2

The second edition of Senseval (Edmonds and Cotton, 2001) was held in Toulouse (France), in July 2001. It was organized under the auspices of ACL-SIGLEX, and the workshop took place just before the main ACL-2001 Conference. For Senseval-2, there were three types of tasks on 12 languages:

- Lexical-sample task: a predefined set of words is chosen, and only instances corresponding to those words are tagged. Most of the languages chose this approach in order to build their tasks.
- All-words task: all content words in a sample of running text are tagged.
- Translation: this is a kind of lexical-sample task where the senses are defined by means of translations to another language. This approach was only applied for Japanese.

A total of 93 systems from 34 groups participated in the different tasks. The majority competed in the English lexical-sample and all-words tasks. As we saw in sectionII.3, the WordNet 1.7 (pre-release) sense inventory was chosen for English.

In the English lexical-sample the best system (JHU) scored 64.2%[10], for 51.2% of the Lesk baseline (described below). Table II.3 shows the results for the lexical-sample task. The position, precision, recall, and coverage of each of the 20 competing systems is given. The organization implemented some baseline systems as reference. These are the more representative: Lesk-corpus (51.2% recall, see previous section for description), MFS (47.6% recall), and Random (14.1% recall).

[10]There was the option of resubmittion to correct some bugs. This decision was adopted because of the tight schedule of the process.

Position	Precision	Recall	Coverage	System
1	64.2	64.2	100.0	JHU (R)
2	63.8	63.8	100.0	SMUls
3	62.9	62.9	100.0	KUNLP
4	61.7	61.7	100.0	Stanford - CS224N
5	61.3	61.3	100.0	Sinequa-LIA - SCT
6	59.4	59.4	100.0	TALP
7	57.1	57.1	100.0	Duluth 3
8	56.8	56.8	99.9	UMD - SST
9	57.3	56.4	98.3	BCU - ehu-dlist-all
10	55.4	55.4	100.0	Duluth 5
11	55.0	55.0	100.0	Duluth C
12	54.2	54.2	100.0	Duluth 4
13	53.9	53.9	100.0	Duluth 2
14	53.4	53.4	100.0	Duluth 1
15	52.3	52.3	100.0	Duluth A
16	50.8	50.8	99.9	Duluth B
17	49.8	49.8	99.9	UNED - LS-T
18	42.1	41.1	97.7	Alicante
19	66.5	24.9	37.4	IRST
20	82.9	23.3	28.0	BCU - ehu-dlist-best

Table II.3: Table of the supervised systems in the Senseval-2 English lexical-sample task sorted by recall (version 1.5, published 28 Sep. 2001). Fine-grained scoring. R: resubmitted system.

The results of this table show that the performance is much lower than in Senseval-1, where the best systems scored in a 75%-78% recall range, and the Lesk baseline reached 69% recall. The main reason for this seems to be the fine-grainedness of the WordNet senses, specially in the case of verbs. As expected, the supervised systems were those performing best. There were some teams that introduced methods from the ML literature for the first time to WSD: AdaBoost (TALP), SVM (UMD-SST), or Maximum Entropy (Alicante). However, the top-scores in this task were for supervised systems that relied on the following characteristics:

- Voting of heterogeneous systems (JHU, Stanford-CS224).
- Rich features: syntactic relations (JHU), Named Entities (SMU), WordNet Semantic Codes (LIA-Sinequa), and WN Domains (TALP).
- Feature selection (SMU) and weighting (JHU).
- Automatically extended training-set (SMU).

JHU and SMUls (which also participated in all-words) will be described in this section after this introduction to Senseval-2. Our own system BCU-ehu-dlist-all[11] is presented in section III.5.8. The unsupervised systems that took part in this task are described in section VI.6.2, where we compare them to our unsupervised methods. The Senseval-2 lexical sample data has been used to test several WSD methods since it was released, we will present some of the most successful (attending to performance) in section V.2.

Regarding the English all-words task, the results are shown in table II.4. The top-scoring methods in the all-words task were also supervised systems, which relied mostly on Semcor for training (SMUaw used also WordNet examples and an automatically generated corpus). We can see that the best system (SMUaw) scored 69%, with a gain of more than 5% over the 2nd system (Ave-Antwerp). A baseline that would assign the 1st sense in WN would score 57%. An indicator of the difficulty of this task is that only 4 out of 21 systems were able to overcome the 1st sense baseline. We will describe the top-3 from the list in the following description of Senseval-2 systems. Our own system (BCU-ehu-dlist-all) is presented in section III.5.8.

JHU (Yarowsky et al.*, 2001)*

This was the best scoring system in the lexical-sample task with 64.2% recall; with an architecture consisting on voting-based classifier combination. A rich set of features was extracted from the context, including syntactic relations (object, subject, noun/adjective modifier, ...) extracted by means of heuristic patterns and regular expressions over the PoS tags around the target word.

Four algorithms were included in the voting ensemble: vector cosine similarity (similar to the VSM described in section II.4.4), Bayesian models (word-based and lemma-based), and DLs. Different voting schemes were tested in cross-validation before submission: probability interpolation, rank-averaged, equal weight, performance-weighted, and thresholded.

SMUls and SMUaw (Mihalcea and Moldovan, 2001)

These systems were applied to the lexical-sample task (ranking 2nd, with 63.8% recall), and the all-words task (winner, with 69% recall). The archi-

[11] We also submitted another system (BCU-ehu-dlist-best), which relied on a precision/coverage threshold, in a fashion similar to the methods we will see in section IV.7.

Position	Precision	Recall	Coverage	System
1	69.0	69.0	100.0	SMUaw
2	63.6	63.6	100.0	CNTS-Antwerp
3	61.8	61.8	100.0	Sinequa-LIA - HMM
4	57.5	56.9	98.9	UNED - AW-U2
5	55.6	55.0	98.9	UNED - AW-U
6	47.5	45.4	95.5	UCLA - gchao2
7	47.4	45.3	95.5	UCLA - gchao3
8	41.6	45.1	108.5	CL Research - DIMAP
9	50.0	44.9	89.7	UCLA - gchao
10	36.0	36.0	99.9	Universiti Sains Malaysia 2
11	74.8	35.7	47.7	IRST
12	34.5	33.8	97.8	Universiti Sains Malaysia 1
13	33.6	33.6	99.9	Universiti Sains Malaysia 3
14	57.2	29.1	50.7	BCU - ehu-dlist-all
15	44.0	20.0	45.3	Sheffield
16	56.6	16.9	29.8	Sussex - sel-ospd
17	54.5	16.9	31.0	Sussex - sel-ospd-ana
18	59.8	14.0	23.3	Sussex - sel
19	32.8	03.8	11.6	IIT 2
20	29.4	03.4	11.6	IIT 3
21	28.7	03.3	11.6	IIT 1

Table II.4: Table of the supervised systems in the Senseval-2 English all-words task sorted by recall (version 1.5, published 28 Sep. 2001). Fine-grained scoring.

tecture has two main components: Instance Based Learning (IBL)[12], when there is specific training data for the target words (lexical-sample task), and pattern learning when there are few examples (all-words task). The system has a pre-processing phase, where Named Entities (NE) and Collocations are detected.

For pattern learning, the examples are obtained from Semcor, WN examples, and GenCor (automatically generated corpora, described in Mihalcea (2002)). The patterns are extracted from the local context of words, and follow the rules of regular expressions, where each token is represented by its base form, its PoS, its sense (when available), and its hypernym (when available). Wildcards (*) are used when the elements are underspecified.

IBL follows the idea of (Veenstra *et al.*, 2000), which participated in Senseval-1 with the "Tilburg" system. In this case, the TiMBL software (Daelemans *et al.*, 2002) is used with information-gain feature weighting. The novelty of this work is that they perform feature selection per each

[12] Also noun as Memory Based Learning (MBL).

word, using cross-validation in training data. Only the features that help to increase performance are kept for each word.

For the lexical-sample task, only IBL was used. Regarding the all-words system, the algorithm followed these steps sequentially until a sense was assigned:

1. Apply IBL when there are enough examples for the word.
2. Apply pattern learning.
3. Propagate senses to close occurrences of the same words in the context.
4. Assign the 1st sense in WordNet.

Ave-Antwerp (Hoste et al.*, 2001)*

The Antwerp all-words system relies on Semcor to build word-experts for each word with more than 10 instances for training. They perform 10 fold cross-validation at 2 levels, in order to optimize the parameters of each of their three classifiers, and also to optimize the voting scheme. Their classifiers consist on 2 versions of their MBL method (TiMBL), trained on different sets of features (local and topical), and a rule learning algorithm called Ripper (Cohen, 1995). Their method scored second in the all-words task, with 63.6% precision and recall.

LIA-Sinequa (Crestan et al.*, 2001)*

This team participated both in the lexical-sample task (ranking in the top-5), and in the all-words task (ranking 3rd). Their all-words system was based on Hidden Markov Models (HMM), trained on Semcor. For the lexical-sample, they relied on Binary Decision Trees trained on the available examples (this was also applied for examples in the all-words task that were also in the lexical sample). The contexts were represented by the lemmas and the WordNet semantic classes in fixed positions around the target word.

II.8 WSD systems in Senseval-3

The third edition of Senseval (Mihalcea and Edmonds, 2004) took place in Barcelona, on July 25-26, 2004, in conjunction with the meeting of the Association for Computational Linguistics (ACL). Fourteen tasks were presented,

System	Team	Precision	Recall
htsa3	University of Bucharest	72.9	72.9
IRST-Kernels	ITC-IRST	72.6	72.6
nusels	National University of Singapore	72.4	72.4
htsa4	University of Bucharest	72.4	72.4
BCU comb	Basque Country University	72.3	72.3
htsa1	University of Bucharest	72.2	72.2
rlsc-comb	University of Bucharest	72.2	72.2
htsa2	University of Bucharest	72.1	72.1
BCU english	Basque Country University	72.0	72.0
rlsc-lin	University of Bucharest	71.8	71.8
HLTC HKUST all	HKUST	71.4	71.4
TALP	U.P. Catalunya	71.3	71.3
MC-WSD	Brown University	71.1	71.1
HLTC HKUST all2	HKUST	70.9	70.9

Table II.5: Top-14 supervised systems in the Senseval-3 lexical-sample task (fine-grained scoring). For each system, the submitting research group and the precision/recall figures are given.

and 55 teams competed on them, for a total of more than 160 system submissions. There were typical WSD tasks (lexical-sample and all-words) for seven languages, and new tasks were included, involving identification of semantic roles, logic forms, multilingual annotations, and subcategorization acquisition. We will focus, as before, on the English lexical-sample and all-words tasks[13].

The English lexical-sample task had the highest participation, as usual. 27 teams submitted 46 systems to this task, most of them supervised. The corpus was built with the collaboration of web users, as is described in section II.3. WordNet 1.7.1 (for nouns and adjectives) and WordSmyth (for verbs) were used as sense inventories. In the official results, 37 systems were considered supervised, and only 9 were unsupervised; but as we mentioned earlier in this chapter, this division is controversial. For instance, it seems clear that the winner in the unsupervised category relied on hand-tagged examples to construct its sense-models. Moreover, the second ranked team acknowledged that their system required a few tagged examples for their clustering method. In any case, the performance of the top-14 supervised systems is given in table II.3[14]. The table shows the name of the system and the submitting team, together with the precision and recall.

The results of the top 14 systems, from 8 different teams, illustrate the

[13]The systems and results in the Basque lexical-sample task are presented in section V.8

[14]Check (Mihalcea *et al.*, 2004) for complete table of supervised methods.

System	Precision	Recall
GAMBL-AW-S	65.1	65.1
SenseLearner-S	65.1	64.2
Koc University-S	64.8	63.9
R2D2: English all-words-S	62.6	62.6
Meaning-allwords-S	62.5	62.3
Meaning-simple-S	61.1	61.0
LCCaw-S	61.4	60.6
upv-shmm-eaw-S	61.6	60.5
UJAEN-S	60.1	58.8
IRTS-DDD-00-U	58.3	58.2

Table II.6: Top-10 systems in the Senseval-3 all-words task. For each system, the precision/recall figures are given.

small differences in performance for this task, where the top-9 systems are less than a point below. This suggests that a plateau has been reached for this kind of task with this kind of ML approaches. The results of the best system (72.9% recall) are way ahead of the MFS baseline (55.2% recall), and present a significant improvement from the previous Senseval edition, which could be due, in part, to the change in the verb sense inventory. Attending to the characteristics of the top-performing systems, this edition has shown a predominance of kernel-based methods (e.g. SVM, see section II.4.5), which have been used by most of the top systems. For instance, the 2nd ranked system works with the kernel function in order to integrate diverse knowledge sources. We will describe the top two systems (Htsa3 and ITC-IRST) in detail below. Other approaches that have been adopted by several systems are the combination of algorithms by voting, and the use of complex features, such as syntactic dependencies and domain tags. Finally, a novelty introduced by the winning system has been a post-processing departure from Bayesian priors, that we will describe below.

Regarding the English all-words task, 20 systems from 16 different teams participated on it. According to the result table presented in (Snyder and Palmer, 2004), 7 systems were supervised and 9 unsupervised (the other four are not categorized). The best system achieved 65.1% precision and recall, while the "WordNet first sense" baseline would achieve 60.9% or 62.4% (depending on the treatment of multiwords and hyphenated words). The results of the top-10 systems are given in table II.4. The suffix (-S) in the name of the system indicates "supervised", and the suffix (-U) indicates unsupervised. Note that the top nine systems are supervised, although the 10th

system (IRTS-DDD-LSI-U), which is a fully-unsupervised domain-driven approach is close to the other methods, and this fact is encouraging for this kind of approach. Furthermore, it is also worth mentioning that in this edition there are more systems above the "first sense" threshold: between four and six.

For the all-words task, there is no plateau, and there are significant differences in the performance and the approaches of the top systems. We will describe the two best-systems (GAMBL-AW and SenseLearner) below. The supervised methods rely mostly in Semcor to get hand-tagged examples; but there are several groups that incorporate other corpora like DSO, WordNet definitions and glosses, all-words and lexical-sample corpora from other Senseval editions, or even the line/serve/hard corpora (Leacock *et al.*, 1998). Most of the participant all-words systems include rich features in their models, specially syntactic dependencies and domain information.

The systems that rank between the 4th and 6th place (R2D2, Meaning-allwords, and Meaning-simple) correspond to collaborative efforts of different research groups, which incorporate supervised and unsupervised approaches in a voting architecture. Kernel-based methods and Domain-driven disambiguation are included in these ensembles. Coincidently, although the ensembles are different, they obtain similar performance.

We will now describe the best performing systems in the English lexical-sample and all-words tasks.

Htsa3 (Grozea, 2004)

The winner in the lexical-sample task was one of the six systems submitted by the group of the University of Bucharest, with 72.9% precision and recall. The learning method applied was Regularized least-squares classification (RLSC), which is based on kernels and Tikhonov regularization. The features that they used consist on local collocations (words, lemmas, and PoS tags), and lemmas in the context of the target word.

Htsa3 relied on a linear kernel, and they normalized its weight-values by dividing them with the empiric frequency of the senses in training data. The normalization helps to balance the implicit bias of RLSC, which gives higher "a posteriori" probability to frequent senses. A new parameter (α) is introduced in order to perform the normalization step smoothly. The regularization parameter and the α value are estimated using the Senseval-1 and Senseval-2 corpora.

IRST-Kernels (Strapparava et al.*, 2004)*

IRST-Kernels scored second in the English lexical-sample task, with 72.6% recall. This system is based on SVM (cf. section II.4.5), and they use the kernel function to combine heterogeneous sources of information. Thus, they define their kernel function as the addition of two kernels: the paradigmatic kernel and the syntagmatic kernel, which are constructed as follows:

- The syntagmatic kernel: the idea is that the similarity between two contexts is given by the shared number of word sequences. This is implemented splitting further the kernel in a "collocation kernel" (based on the lemma sequences) and a "PoS kernel" (based on PoS sequences). In order to include matches of equivalent terms, a similarity threshold based on LSA is applied, and terms above the threshold are considered equal.

- The paradigmatic kernel: information about the domain of the text is introduced by this measure. This kernel is also the addition of another two: a "bag of words" kernel and an "Latent Semantic Indexing (LSI) kernel". The second tries to alleviate the sparseness problem of the "bag of words" kernel.

They conclude that syntagmatic and paradigmatic information are complementary, and they claim that kernels provide a flexible way to integrate different sources of knowledge.

GAMBL-AW (Decadt et al.*, 2004)*

This system was the winner of the all-words task. They submitted a similar system also to the lexical-sample task, which scored lower than kernel-based methods. GAMBL-AW is a supervised approach that relies on extensive corpora to learn the word-experts. This corpus is obtained joining Semcor with all the tagged data from previous Senseval editions (all-words and lexical-sample; training and testing), also including the training data in Senseval-3 lexical-sample, the examples in WordNet, and the line/hard/serve corpora. From these examples, they extract two types of features: the local context (including information about chunks and dependency relations extracted from a shallow parser), and the keywords in context. The keywords are extracted per each sense from two sources: WordNet sense definitions, and applying the method in (Ng and Lee, 1996).

Using this information, GAMBL applies a word-expert approach with MBL (using TiMBL) and optimization of features and parameters. They apply a cascaded architecture, where classification is carried out in two steps: first, a keyword-based classifier assigns a sense to the new example; this sense is then used as a feature for a second classifier, which is based on local features and makes the final decision. In order to construct these two classifiers, exhaustive optimization is performed with Genetic Algorithms (GA) and heuristic optimization by means of cross-validation. They use GAs to jointly optimize feature selection and parameter optimization. They show a significative improvement in the results due to optimization.

SenseLearner (Mihalcea and Faruque, 2004)

SenseLearner obtained the 2nd best score in the English all-words task, with 64.2% recall. This team considers one of their goals to use as few hand-tagged data as possible, and they rely only on Semcor and the WordNet hierarchy to construct their architecture. The method applies two main steps sequentially, jumping to the second only when the first abstains:

1. Semantic Language Model: The examples in Semcor are used to learn a model for each PoS (using jointly all the words), based on very simple co-occurrence features, which are different for each PoS. TiMBL is then applied to the testing examples, and the model predicts the word and sense of the test example. If the predicted word corresponds to the example, the predicted sense is assigned, otherwise there is no answer. The average coverage of this method is 85.6%.

2. Semantic Generalizations using Syntactic Dependencies and WordNet: In the learning phase, all the dependencies in Semcor are extracted and expanded with the hypernyms of the nouns and verbs appearing in them. For each dependency-pair, positive feature vectors are created for the occurring senses, and negative vectors for the others. In the testing phase, for each dependency-pair, feature vectors are created for all possible combinations of senses. TiMBL assigns a positive or negative value for each of this vectors, using the generalizations extracted from Semcor. These values are used to make the final prediction.

III. CHAPTER

Baseline WSD system: DL and basic features

III.1 Introduction

In the previous chapter we have seen some supervised WSD techniques, and the state-of-the-art performance that different systems can provide. In order to approach the main problems of a complex phenomena like WSD (issues introduced in the first chapter), we will now implement our own baseline disambiguation system. The idea when constructing this system is to use basic resources (WordNet, publicly available corpora, well-studied feature types and algorithms, etc.) and apply them to our experiments. Our goal in this chapter is twofold:

1. Apply our basic system to extensive experimentation in order to shed light into different aspects of WSD.

2. Measure the performance we can obtain with this system to be used as reference when we introduce improvements, like new knowledge sources, disambiguation algorithms, or automatically acquired examples.

As we mentioned in the introduction chapter, the DL algorithm (c.f. section II.4.2) has some qualities that make it a good candidate for our basic system:

- DLs are based on the best single evidence, in opposition to classification based on the combination of contextual evidences. Therefore, multiple

non-independent features can be included in the system without having to model the dependencies.

- The decision lists built for each word can be hand-inspected, and provide useful information about the target word.

- Despite its simplicity, this algorithm has performed well in different Senseval editions; as single system in Senseval-1, and as part of an ensemble in Senseval-2.

Thus, we will use DLs as learning method, and we will test how far can we go with existing hand-tagged corpora (cf. section II.3) like Semcor, the DSO corpus, and the Senseval-2 data, which have been tagged with word senses from WordNet. The feature-types that are used for disambiguation will be one of the topics of this dissertation. Throughout this chapter, we will rely on a basic set of features, similar to those widely used for WSD in the literature (Yarowsky, 1994; Ng and Lee, 1996), which we will separate into local and topical sets. The separation into two main sets will allow us to start analyzing the effect of feature-types for disambiguation performance.

Now that we have presented our baseline system, we will perform a precision/coverage evaluation on this setting, and we will also address some questions that we consider relevant about supervised WSD:

1. Word types: relation between polisemy/bias/frequency and performance.

2. Feature types: relation between word types and basic feature types.

3. How much data is needed? Learning curve.

4. How much noise in the data can be acceptable?

5. Fine-grained vs. coarse-grained disambiguation.

6. Expected performance for all words in a text.

7. Comparison with other methods in a real setting: Senseval-2.

8. Study performance for another language, less studied and with less resources: Basque.

We expect these experiments to give us more insight into the problem, before we start focusing on the main contributions of this work.

The remaining of this chapter is organized as follows. Following this introduction, we will introduce briefly works in the literature that are related to the experiments we present in this chapter. The next section will be dedicated to the experimental settings that we will apply in this and other chapters of this dissertation. In the following section, we will describe the extraction of basic features from the context, for English and Basque. After that, the main section will be devoted to the experiments that try to shed light on the questions presented above, devoting one section to each. Finally, some conclusions will be outlined.

III.2 Related work

The experiments that we will carry out in this chapter present different aspects of the WSD problem, which in some cases have been studied in the literature. As the experiments cover diverse works, we decided to introduce them briefly here, and describe them in more detail in the sections corresponding to the experiments.

Our first reference to the WSD literature will come with the study of local and topical features (cf. section III.5.3), where we will compare our results with those reported in (Gale *et al.*, 1993) and (Leacock *et al.*, 1998). In order to justify the different conclusions working on Semcor or DSO, we will refer to the work by Ng *et al.* (1999). Also in this section, we will recall the work in (Hoste *et al.*, 2002) on the construction of word-experts with tailored feature sets. Regarding the study of the learning curves, in section III.5.4 we will describe the work carried out in (Ng, 1997) on the DSO corpus.

Finally, we will refer to related works to compare the performance of DLs with other algorithms in the same setting. In section III.5.7, we will present (Escudero *et al.*, 2000b), where three ML methods (NB, AB, and K-nearest neighbors) are applied to the disambiguation of all the words in DSO. We will show the results of each algorithm, and compare them to our baseline setting. The conclusions of recent works that include other ML algorithms are also mentioned in this section, with comments on the relative performance of DLs (Yarowsky and Florian, 2002; Villarejo *et al.*, 2004), and on the fluctuations found on the experiments in relation to the parameter space (Yarowsky and Florian, 2002; Hoste *et al.*, 2002). For more

comparative results, sections III.5.8 and III.5.9 describe the systems that we built for the Senseval-2 competition for English and Basque respectively, and show the overall result tables. The English results were previously introduced in section II.7, with the description of some systems.

III.3 Experimental settings

This section is organized in four parts, and will introduce the experimental setting that we will use through the dissertation. First, we will define the target sets for our experiments. Those sets will consist on word and file sets defined from Semcor and DSO, and the datasets in the Senseval tasks. The next subsection will describe the pre-processing of multiwords, specially in relation to the Senseval-2 corpora. The next segment will be devoted to enumerate the different settings that will be used throughout the dissertation, and this subsection will be referenced in the "experimental setting" sections of the chapters to come. Finally, we will outline the setting that will be applied for the work in this chapter.

III.3.1 Test sets

In order to evaluate our system, we have to choose a target word set. We can choose a fixed set of "representative" words, or we can take a corpus and try to disambiguate all the words that appear. For our first experiments in Semcor and DSO we selected a set of words attending to criteria like frequency, ambiguity and bias. We also disambiguated all the content words in some given files, and all the DSO corpus. For the different Senseval tasks, we used as test words the ones provided by the organization.

III.3.1.1 Semcor test set

We selected 19 test words trying to cover the maximum variety of cases. Thus, we classified them according to these factors:

- Frequency: number of training examples in Semcor (low, high)
- Ambiguity: number of senses (low, high)
- Bias: skew of most frequent sense in Semcor (low, high)

As we will see in section III.5.2, the two first criteria are interrelated (frequent words tend to be highly ambiguous), but there are exceptions. The third criterion seems to be independent, but high bias is sometimes related to low ambiguity. We could not find all 8 combinations for all parts of speech and the following samples were selected: 2 adjectives, 2 adverbs, 8 nouns and 7 verbs. These 19 words form the test set A. The DSO corpus does not contain adjectives or adverbs, and focuses in high frequency words. Only 5 nouns and 3 verbs from Set A were present in the DSO corpus, forming Set B of test words. The list of words can be consulted in the appendix (cf. table B.6).

In addition, 4 files from Semcor previously used in the literature (Agirre and Rigau, 1996) were selected, and all the content words in the files were disambiguated.

III.3.1.2 DSO test set

Another word-set was defined for the experiments that relied on the DSO corpus, as the previously defined set B contained only 8 words. In this case we used a set of 21 verbs and nouns previously used in the literature (Agirre and Martinez, 2000; Escudero *et al.*, 2000c). We will refer to these words as set C. The list of words is given in the appendix (cf. table B.6).

III.3.1.3 Senseval test sets

The different senseval tasks provide different word-sets and contexts to evaluate the systems. This is the list of tasks our systems have been tested on (the tables with the words in the lexical-sample tasks are given in section B.1 in the appendix):

- Senseval-2 English lexical-sample task (73 words)
- Senseval-2 Basque lexical-sample task (40 words)
- Senseval-3 English lexical-sample task (57 words)
- Senseval-3 Basque lexical-sample task: (40 words)
- Senseval-2 English all-words task

III.3.2 Pre-process of multiwords

In some cases the target word we want to disambiguate is part of a multiword that has its own entry in a sense repository. In WordNet, for example, many multiwords are represented, e.g. *church building*, *fine arts*, etc. For lexical-sample tasks, in some cases the multiwords are excluded (English tasks in Senseval-1 and Senseval-3), and in other cases they have to be detected (Basque tasks in Senseval-2 and Senseval-3, English task in Senseval-2). In order to recognize multiword senses, they can be included in the sense-list of the target word and treated like other senses. However, usually a better option is to incorporate a pre-processing stage to try to detect them with a search in the context.

Regarding our experiments, in the all-words corpora we used (Semcor, Senseval-2 and Senseval-3), multiwords are always marked; although in the case of Senseval-2 it was not always easy to identify them, and this could affect our results (cf. section III.5.8). For the lexical-sample tasks, we adopted three approaches in different experiments:

1. Treat the multiword senses as any other sense: this approach was adopted for our first experiments with the English task in Senseval-2[1] (sections III.5.8, IV.6, and IV.7), and also for the Basque tasks (sections III.5.9 and V.8).

2. Remove multiword senses (and proper nouns): we chose this setting in order to avoid noise in our experiments on automatic acquisition of examples with the Senseval-2 English data (chapter VI).

3. Apply a pre-process to detect multiword senses: this step was integrated for the experiments in chapter V with the Senseval-2 English data, and is explained below.

In order to achieve better performance in lexical-sample settings with multiword senses, we built a supervised tool to detect them independently. The tool proceeds by identifying all the lemmas around the target word that appear in WordNet (continuous and non-continuous), and using DLs learned from training data for the specific ambiguity (e.g. to determine whether *art*

[1] Many of the multiword cases in Senseval-2 were phrasal verbs. The Senseval-2 corpus also included proper-noun marks as sense tags, these cases were discarded due to their difficulty.

or *arts* is the correct lemma, only examples from training that have those candidates are used). The training data is usually scarce, but the recall of this process reaches 96.7% in the Senseval-2 English lexical-sample corpus.

III.3.3 Specific Settings

In our study of different aspects of WSD, we will apply different settings (corpora, sense-inventories, and word sets) depending on the parameters we are studying and the resources available at the moment. In this section we will describe the main settings that will be applied throughout the experiments.

III.3.3.1 Semcor&DSO

This setting has been used for the basic set of experiments (chapter III), and for experiments with richer features (chapter IV). See figure III.1.

III.3.3.2 WSJ&BC

We have applied this setting in order to study genre/topic variations (chapter VII). The main characteristics are given in figure III.2.

III.3.3.3 Senseval2

This setting is related to the three tasks on which our systems participated in the Senseval-2 competition: English lexical-sample task, English all-words task, and Basque lexical-sample task. This setting is frequently applied, specifically in chapter III, chapter IV, and chapter V. See figure III.3 for details on this setting.

III.3.3.4 Senseval2B

For this setting, the multiword senses that appear in the Senseval-2 lexical-sample English corpus are removed, in order to test the automatic acquisition of sense-tagged examples (chapter VI). The results training on the automatically obtained examples are compared with the results training on Semcor. Figure III.4 shows the characteristics of this setting.

III.3.3.5 Senseval3

This setting is related to the two tasks on which our systems participated in the Senseval-3 competition: English lexical-sample task, and Basque lexical-sample task. This setting is applied in chapter V. Details are given in figure III.5.

- Corpora: Semcor and DSO.
  - Used separately, by applying cross-validation in each corpus.
- Sense inventories:
  - WordNet 1.6 for experiments in Semcor.
  - WordNet 1.5 for experiments in DSO.
- Word-sets:
  - set A and set B.
  - All words in 4 Semcor files.
  - All 191 tagged words in DSO.

Figure III.1: Semcor&DSO setting.

- Corpora: DSO.
  - The two parts of DSO (WSJ and BC) used separately for cross-tagging.
  - Cross-validation to evaluate each corpus separately.
- Sense inventory: WordNet 1.5.
- Word-set: C.

Figure III.2: WSJ&BC setting.

- Corpora:
  - Lexical tasks:
    - Senseval-2 English lexical-sample corpus: separated training and test.
    - Senseval-2 Basque lexical-sample corpus: separated training and test.
  - All-words task (English):
    - Semcor for training.
    - Senseval-2 English all-words task corpus for testing.
- Sense inventories:
  - WordNet 1.7Pre: English lexical-sample and all-words corpora.
  - WordNet 1.6: Semcor (automatically mapped to WN 1.7Pre (Daude *et al.*, 2000)).
  - EH inventory: sense list from a Basque dictionary (cf. section II.2).
- Word-sets: Target sets in Senseval-2 (cf. sections III.3.1.3 and B.1).

Figure III.3: Senseval2 setting.

- Corpora:
  - Training: Semcor and automatically retrieved examples.
  - Testing: Subset of Senseval-2 English lexical-sample testing part.
    - Examples tagged with multiword senses, phrasal verbs, and proper nouns removed (cf. section II.3).
- Sense inventories:
  - WordNet 1.7Pre: English lexical-sample testing and automatically retrieved examples.
  - WordNet 1.6: Semcor (automatically mapped to WN 1.7Pre (Daude *et al.*, 2000)).
- Word-set: The nouns in the Senseval-2 English lexical-sample task.

Figure III.4: Senseval2B setting.

- Corpora:
  - Senseval-3 English lexical-task corpus: separated training and test.
  - Senseval-3 Basque lexical-task corpus: separated training and test.
- Sense inventories:
  - Nouns and adjectives are annotated using the WordNet 1.7.1 sense inventory (http://www.cogsci.princeton.edu/ wn/).
  - Verbs are annotated based on Wordsmyth definitions (http://www.wordsmyth.net)
- Word-sets: Target sets in Senseval-3 (cf. sections III.3.1.3 and B.1).

Figure III.5: Senseval3 setting.

III.3.4 Experimental setting for this chapter

Most of the experiments performed in this chapter will use the Semcor and DSO corpora, in the setting described in section III.3.3.1. As features, the basic feature sets introduced in the following section III.4 will be applied. Only the last experiments that compare our system to others in the Senseval-2 framework will require a different setting (Senseval-2 setting, cf. section III.3.3.3). In this case also the feature set has slight differences for English, and this will be explained in the corresponding section.

III.4 Extraction of features from the context

This section is devoted to the extraction of basic features. We will first describe the feature-set we will use for the baseline English system. After that, we will explain the process to construct the feature-set that will be applied in our experiments with Basque.

III.4.1 Basic features for English

Throughout the thesis, we will use different feature sets, and different tools to extract them from the context. For English, there will be different features depending on the phenomena we are studying, and these will be described

more accurately in the corresponding sections. In this part, we will introduce the basic feature set that we will apply in chapter III.

We have taken as basic feature set a group of items from the context widely used in the literature (Yarowsky, 1994; Ng and Lee, 1996). We will separate them into topical and local features.

Topical features correspond to open-class word-forms that appear in windows of different sizes around the target word. In this experiments we used two different window-sizes: 4 words around the target, and the word-forms in the sentence.

Local features include bigrams and trigrams that contain the target word. Local features are formed by the PoS, or word-forms. Lemmatization is not used in this basic feature set, because we will test it separately, together with more informed features.

We can see an example of the extraction of features in figure III.6. Each line corresponds to one feature. The analysis of the raw text to obtain lemmas and PoS tags is done differently depending on the corpus. Semcor provides all the information, including multiwords and named entities; DSO and Senseval-2 were processed using different tools. The PoS tagging was performed with TnT (Brants, 2000) for the basic set of features. For posterior experiments, the fnTBL toolkit (Ngai and Florian, 2001) was applied. For the lemmatization, we used the functions provided with the WordNet distributions.

III.4.2 Basic features for Basque

For Basque we only apply a set of features, which will be described in this section, along with the main characteristics of the language. It would be interesting to analyze other possibilities, but that study is out of the scope of this thesis.

Basque is an agglutinative language, and syntactic information is given by inflectional suffixes. The morphological analysis of the text is a necessary previous step in order to select informative features. We used the output of the parser (Aduriz *et al.*, 2000), which includes some additional features: number, determiner mark, ambiguous analyses and elliptic words. For a few examples, the morphological analysis was not available, due to parsing errors.

In Basque, the determiner, the number and the declension case are appended to the last element of the phrase. When defining our feature set for Basque, we tried to introduce the same knowledge that is represented by

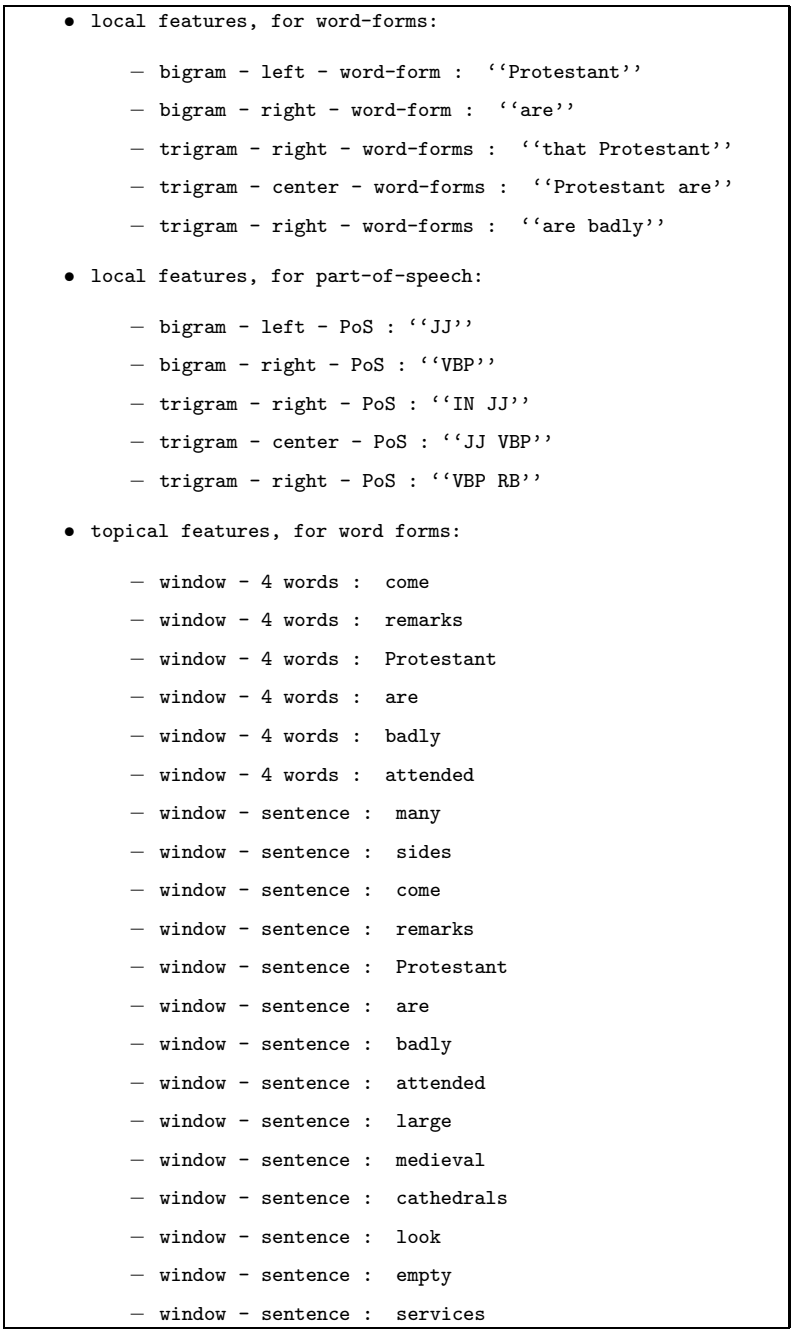

Figure III.6: Features extracted for the target word *church* from the sentence *From many sides come remarks that Protestant* ***churches*** *are badly attended and the large medieval cathedrals look all but empty during services.*

features that work well for English. We will describe our feature set with an example. For the phrase "elizaren arduradunei" (which means "to the directors of the church") we get the following analysis from our analyzer:

eliza	|-ren	|**arduradun**	|-ei
church	|of the	|**director**	|to the +plural

The order of the words is the inverse in English. We extract the following information for each word:

```
elizaren:

Lemma:  eliza (church)
PoS: noun
Declension Case:  genitive (of)
Number:  singular
Determiner mark:  yes

arduradunei:

Lemma:  arduradun (director)
PoS: noun
Declension Case:  dative (to)
Number:  plural
Determiner mark:  yes
```

We will assume that eliza (church) is the target word. Words and lemmas are shown in lowercase and the other information in uppercase. As local features we defined different types of unigrams, bigrams, trigrams and a window of ± 4 words. The unigrams were constructed combining word forms, lemmas, case, number, and determiner mark. We defined 4 kinds of unigrams:

```
Uni_wf0 elizaren
Uni_wf1 eliza SING+DET
Uni_wf2 eliza GENITIVE
Uni_wf3 eliza SING+DET GENITIVE
```

As for English, we defined bigrams based on word forms, lemmas and parts-of-speech. But in order to simulate the bigrams and trigrams used for English, we defined different kinds of features. For word forms, we distinguished two cases: using the text string (Big_wf0), or using the tags from the analysis (Big_wf1). The word form bigrams for the example are shown below. In the case of the feature type "Big_wf1", the information is split in three features:

```
Big_wf0 elizaren arduradunei
Big_wf1 eliza GENITIVE
Big_wf1 GENITIVE arduradun_PLUR+DET
Big_wf1 arduradun_PLUR+DET DATIVE
```

Similarly, depending on the use of the declension case, we defined three kinds of bigrams based on lemmas:

```
Big_lem0 eliza arduradun
Big_lem1 eliza GENITIVE
Big_lem1 GENITIVE arduradun
Big_lem1 arduradun DATIVE
Big_lem2 eliza_GENITIVE
Big_lem2 arduradun_DATIVE
```

The bigrams constructed using Part-of-speech are illustrated below. We included the declension case as if it was another PoS:

```
Big_pos_-1 NOUN GENITIVE
Big_pos_-1 GENITIVE NOUN
Big_pos_-1 NOUN DATIVE
```

Trigrams are built similarly, by combining the information from three consecutive words. We also used as local features all the content words in a window of ± 4 words around the target. Finally, as global features we took all the content lemmas appearing in the context, which was constituted by the target sentence and the two previous and posterior sentences.

One difficult case to model in Basque is the ellipsis. For example, the word "elizakoa" means "the one from the church". We were able to extract this information from our analyzer and we represented it in the features, using a special symbol in place of the omitted word.

III.5 Experiments

Now that we have introduced the experimental setting, this section is devoted to the study of the main questions raised in the introduction. The experiments will explore different aspects of the WSD problem.

III.5.1 Baseline and basic features

In our first experiment, we used the basic feature set defined in section III.4.1 to train the system, and compared the results with two baselines: the random baseline and the more informed MFS baseline (cf. section II.4.1). The random baseline is directly obtained by means of the ratio of the total number of senses. The experiment was performed for the 19 words in set A (for Semcor), and the 8 words in set B (for DSO).

Word	PoS	S.	Rand	Semcor				DSO			
				Ex.	Ex./S	MFS	DL	Ex.	Ex./S	MFS	DL
All	A	2	50	211	105.50	**99**	**99**/100				
Long	A	10	10	193	19.30	53	**63**/99				
Most	B	3	33	238	79.33	74	**78**/100				
Only	B	7	14	499	71.29	51	**69**/100				
Account	N	10	10	27	2.70	44	**57**/85				
Age	N	5	20	104	20.80	72	**76**/100	491	98.20	62	**73**/100
Church	N	3	33	128	42.67	41	**69**/100	370	123.33	62	**71**/100
Duty	N	3	33	25	8.33	32	**61**/92				
Head	N	30	3	179	5.97	78	**88**/100	866	28.87	40	**79**/100
Interest	N	7	14	140	20.00	41	**62**/97	1479	211.29	46	**62**/100
Member	N	5	20	74	14.80	**91**	**91**/100	1430	286.00	74	**79**/100
People	N	4	25	282	70.50	**90**	**90**/100				
Die	V	11	9	74	6.73	**97**	**97**/99				
Fall	V	32	3	52	1.63	13	**34**/71	1408	44.00	75	**80**/100
Give	V	45	2	372	8.27	22	**34**/78	1262	28.04	75	**77**/100
Include	V	4	25	144	36.00	**72**	70/99				
Know	V	11	9	514	46.73	59	**61**/100	1441	131.00	36	**46**/98
Seek	V	5	20	46	9.20	48	**62**/89				
Understand	V	5	20	84	16.80	**77**	**77**/100				

Table III.1: Information for the words in set A (Semcor) and set B (DSO), and results for baselines (Random and MFS) and DL (trained with the basic set of features).
S: number of senses; Rand: Random baseline; Ex./S: number of examples per sense.

The results for the Semcor and DSO corpus for each word are shown in table III.1. For DLs the precision and coverage are given, for the baselines only the precision (the coverage is always 100%). Complementing the precision and coverage figures, the following information is provided for each word: number of senses in WordNet 1.6, number of examples in the corpus, and number of examples per sense (frequency/ambiguity ratio). These figures can give an idea of the difficulty of the words. E.g. *fall* only has 1.63 examples per sense, and the MFS precision is 13%, which indicates that we should not expect high accuracy. For this word, DLs obtain 34% precision for 71% coverage, showing that the system is able to achieve results over the MFS baseline even with few training data.

We have marked the winning column in boldface, and we can see that DLs beat the baselines almost in all cases in Semcor (only *include* gets slightly lower results than MFS). In DSO, DLs are always better than the baselines. With respect to the coverage, for Semcor it does not reach 100% in all cases,

	Semcor								**DSO**		
	set A					**set B**			**set B**		
	Adj.	**Adv.**	**Noun**	**Verb**	**Over.**	**Noun**	**Verb**	**Over.**	**Noun**	**Verb**	**Over.**
Senses	5.8	5.7	9.4	20.3	12.3	10.0	29.3	17.2	10.1	28.6	18.8
Examples	202	368.5	119.9	183.7	178.2	125.0	312.7	195.4	927.2	1370.3	1093.4
Examples per sense	34.7	64.5	12.6	9.0	14.4	12.5	10.7	11.3	92.7	46.7	63.4
Random	31	20	19	10	17	16	6	10	16	5	1
MFS	77	58	69	51	61	63	42	50	56	61	59
DL	**82/** 100	**72/** 100	**80/** 99	**58/** 92	**70/** 97	**77/** 99	**49/** 90	**60/** 94	**72/** 100	**67/** 99	**70/** 100

Table III.2: Average results (DL and baseline), and statistics for the basic set of features in Semcor and DSO. For DLs, precision and coverage are given.

because some decisions are rejected when the log likelihood is below zero. On the contrary, the richer data in DSO enables 100% coverage.

For a better analysis, table III.2 groups the previous values per word-set and PoS. The values are micro-averaged with the number of examples per word (cf. section II.5.1). The three upper rows of the table illustrate the relation between the ambiguity and the frequency for the word-sets in Semcor and DSO: average number of senses, examples, and examples per sense (ratio). The next two rows indicate the precision of the random and MFS baselines (always with full coverage). Finally, the performance of DLs is shown (precision and coverage given). The best precision for each column is denoted in boldface. We will point out some conclusions from this table:

- The number of examples per word sense is very low for Semcor (around 11 examples per sense for the words in Set B), while DSO has substantially more training data (around 66 examples per sense in set B). It has to be noted that several word senses do not occur neither in Semcor nor in DSO.

- The random baseline attains 17% precision for Set A, and 10% precision for Set B.

- The MFS baseline is higher for the DSO corpus (59% for Set B) than for the Semcor corpus (50% for Set B). This rather high discrepancy can be due to tagging disagreement, as will be commented in the concluding section of the chapter.

- The scarce data in Semcor seems enough to get results over the baselines. The larger amount of data in DSO warrants a better performance, but it is still limited to 70% precision. Overall, DLs significantly outperform the two baselines in both corpora:
  - Set A: 70% vs. 61% (Semcor).
  - Set B: 60% vs. 50% (Semcor), 70% vs. 59% (DSO).
- If we analyze the words according to PoS, we can clearly see that verbs get the lowest precision, specially in Semcor (58% for set A, 49% for set B). Verbs are very ambiguous, and the "examples per sense" ratio is low. In DSO, the difference in precision for verbs and nouns is not so evident (72% for nouns, 67% for verbs), even when the example ratio for nouns is twice as high (92 to 46). Nouns, adjectives, and adverbs always score over 70%, adjectives reaching 82% precision in Semcor. The worst coverage is also for verbs: in Semcor 92% of the examples are covered for set A and 90% for set B; while for the other PoS the coverage is 99% or 100%.
- Looking at the difference between DLs and MFS, we notice that verbs get the lowest improvement in almost all cases, except for the adjectives in Semcor, which have a high MFS (77%), difficult to beat. The most impressive gain is for adverbs, which improve MFS in 14 points. In section III.5.2, we will study words according to their polisemy, frequency, and skew; and we will extract some conclusions in relation to the performance over the baselines.
- It is difficult to compare the results obtained for set B in Semcor and DSO, probably due to the discrepancies tagging the same words on different corpora. By PoS, we can see that the precision is much better for verbs in DSO (67% vs. 49%), but for nouns it is better in Semcor (77% vs. 72%). Even if DSO has much more training data (927 examples per word in average for nouns, versus 125 examples per word in Semcor), and we would expect the precision to be higher. We can see in table III.1 that 4 out of 5 nouns achieve better results on Semcor data. The reason could be that, even if the word set is the same, the tagging differences make the task different.

Word	POS	Examples	Testing time (secs.)	Preprocessing and training time (secs.)
All	A	211	2.00	711.20
Long	A	193	2.00	745.20
Most	B	238	2.40	851.80
Only	B	499	5.20	1143.50
Account	N	27	0.00	131.60
Age	N	104	1.00	302.90
Church	N	128	1.00	175.60
Duty	N	25	0.00	133.30
Head	N	179	1.20	500.40
Interest	N	140	1.30	397.20
Member	N	74	1.00	303.70
People	N	282	2.80	686.60
Die	V	74	0.20	276.50
Fall	V	52	0.20	303.10
Give	V	372	4.60	968.30
Include	V	144	1.30	526.70
Know	V	514	4.40	924.30
Seek	V	46	0.00	230.80
Understand	V	84	0.90	344.70
set A	Avg. A	202.00	2.00	728.20
	Avg. B	368.50	3.80	997.65
	Avg. N	119.88	1.04	328.91
	Avg. V	183.71	1.66	510.63

Table III.3: Execution time for DL with the examples in Semcor.

Regarding the execution time, table III.3 shows training and testing times for each word in Semcor. Training the 19 words in set A takes around 2 hours and 30 minutes, and it is linear to the number of training examples, around 2.85 seconds per example. Most of the training time is spent processing the text files and extracting all the features, which includes complex window processing. Once the features have been extracted, training time is negligible as also is the test time (around 2 seconds for all instances of a word). Training time has been measured on CPU total time on a Sun Sparc 10 machine with 512 Megabytes of memory at 360 Mhz.

III.5.2 Kinds of words: polisemy/bias/frequency

In this experiment we analyzed the effect on disambiguation performance of three factors: ambiguity, frequency, and bias. These characteristics have been defined in section III.3.1.1, and have been used to choose the word-sets. Our goal was to observe whether the disambiguation precision of a word can be determined by its ambiguity, frequency, or bias. We measured the absolute precision of DLs, and the precision of the DLs relative to the

baselines (difference in precision).

We developed the experiment with the set A of words, and the Semcor corpus. We used the results on the previous experiment (see table III.1) to draw the precision depending on the different factors, and study them. These are our conclusions:

- Frequency: Figure III.7 plots the precision (absolute and relative) according to the number of examples employed to train each word. The resulting graph shows no improvement for words with larger training data. This is due to the interrelation between frequency and ambiguity. The words that are more ambiguous, have more examples in Semcor. Therefore, the frequency of a word in the corpus does not determine the disambiguation precision we can expect.

- Ambiguity: Similarly, the data of figure III.8 does not indicate whether ambiguous words are easier to disambiguate. Again, the reason is that words with many senses occur more frequently.

- Bias: This is the parameter that affects the performance the most. Words with high skew obtain better results, but the DLs outperform MFS mostly on words with low skew (see figure III.9).

Overall decision lists perform very well (compared to MFS) even with words with very few examples (E.g.: *duty* (25) or *account* (27)) or highly ambiguous words.

III.5.3 Feature types: relation between word types and basic feature types

The goal of this experiment was to analyze separately the performance of features from local and global contexts in disambiguation. There have been previous experiments in this line (Gale *et al.*, 1993; Leacock *et al.*, 1998) and they have shown that topical contexts tend to work better for nouns. For our experiment, we consider bigrams and trigrams (PoS tags and word-forms) as local, and two features as topical: all the word-forms in the sentence, and a four-word window around the target.

The results are illustrated in table III.4. We show the performance achieved for each word, and the micro-averaged results per PoS and overall. In each column, the precision and coverage for each feature set (local features, topical features, combination) is given.

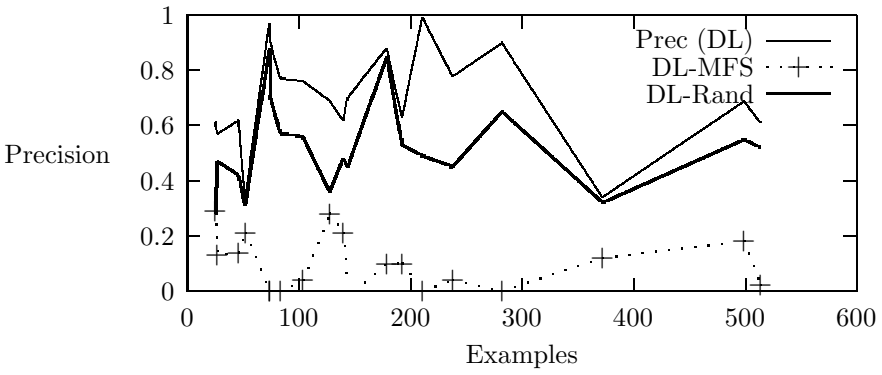

Figure III.7: Results in Semcor according to frequency.

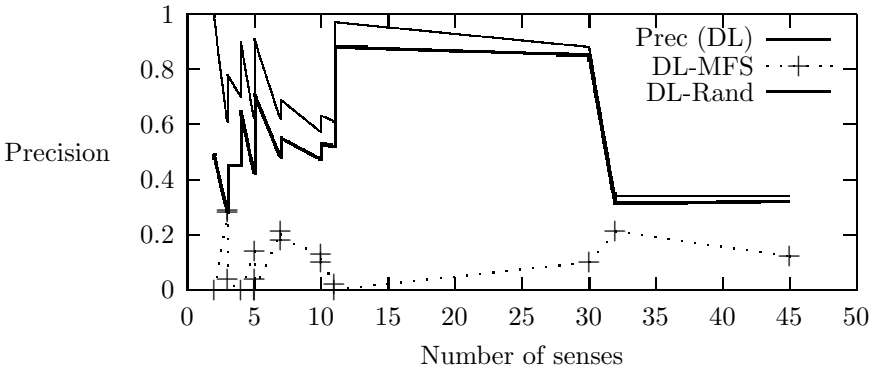

Figure III.8: Results in Semcor according to ambiguity.

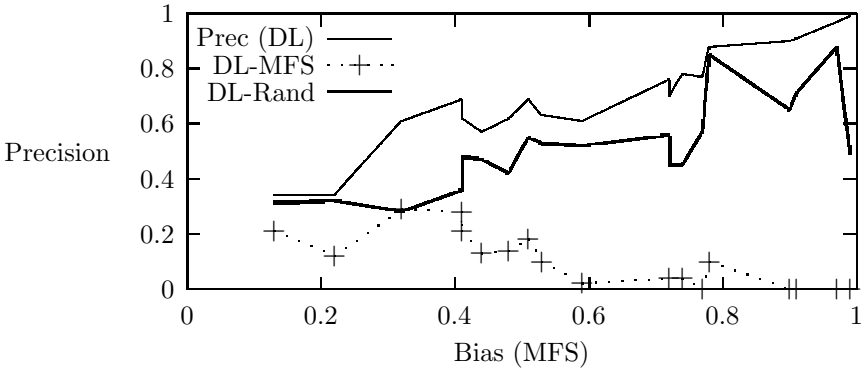

Figure III.9: Results in Semcor according to bias.

Word	Pos	Semcor			DSO		
		Local	Topical	Combination	Local	Topical	Combination
All	A	99/100	98/91	99/100			
Long	A	67/98	61/87	63/99			
most	B	79/100	71/95	78/100			
only	B	72/100	60/96	69/100			
account	N	55/78	47/56	57/85			
age	N	73/99	78/87	76/100	76/98	70/97	73/100
church	N	60/98	74/89	69/100	68/100	72/96	71/100
Duty	N	62/84	75/48	61/92			
Head	N	89/100	90/85	88/100	78/99	76/97	79/100
Interest	N	55/86	57/86	62/97	68/91	60/98	62/100
Member	N	90/99	91/89	91/100	81/100	78/100	79/100
People	N	90/100	89/94	90/100			
Die	V	97/99	96/70	97/99			
Fall	V	35/60	35/25	34/71	81/99	80/96	80/100
Give	V	41/54	32/52	34/78	77/100	78/98	77/100
Include	V	69/98	73/85	70/99			
Know	V	59/99	57/90	61/100	52/89	37/81	46/98
Seek	V	70/80	40/43	62/89			
Understand	V	77/100	75/81	77/100			
Avg. A		84/99	81/89	82/100			
Avg. B		74/100	64/96*	72/100			
Avg. N		78/96	81/87	80/99	75/97	71/98*	72/100*
Avg. V		61/84	57/72	58/92	70/96	66/91*	67/99*
Overall		72/93	68/84*	70/97	73/96	69/95*	70/100*

Table III.4: Local context vs. topical context in Semcor and DSO. Precision and coverage is shown. The mark '*' indicates statistical significance according to the t-test (only for PoS and overall figures).

We also apply a paired Student's t-test of significance (cf. section II.5.2) to see whether the difference between the approaches is statistically significant. We measured this value for local features vs. topical features, and also for the winning system (local or global) vs. combination of features. The confidence value we used was $t_{9,0.975} = 2.262$. We show the results of the test grouped by PoS and overall, not by word; the "topical" and "combination" columns include the character '*' when the t-test indicates that the difference is statistically significant.

The results in Semcor show that, attending to precision, topical features achieve the best results for nouns, while for the other parts of speech the best precision is for the local features. These results are consistent with those obtained by Gale *et al.* (1993) and Leacock *et al.* (1998). Word by word, we can see that 6 out of the 8 nouns obtain better results when trained on topical features, and 12 of the other 13 words learn better from local features.

However, according to the t-test, there is only significant difference in the case of adverbs, and overall. The overall results show that local features get better precision and coverage in comparison to topical features. The combination of all features gets similar recall to local features (68%).

The results with the DSO corpus are in clear contradiction with those from Semcor: local features have better precision both for nouns and verbs. Out of the 8 words, only the noun *church* and the verb *give* achieve better precision with topical features. It is hard to explain the reasons for this contradiction, but it can be related to the amount of data available in DSO, and the differences of tagging in both corpora (Ng *et al.*, 1999). We have to note that for nouns, topical features get better coverage than local features (exactly the opposite behavior to the Semcor results). This could mean that the large amount of data in DSO provides more overlapping cases for the topical features, and those are applied with less confidence, damaging precision. Again, the combination of both kinds of features attains lower precision in average than the local features alone, but this is compensated by a higher coverage, and overall the recall is very similar (70%). The t-test finds the local features significantly better than the topical features for all parts of speech, and the difference between local and combined features is also found to be significant (even if they have similar recall overall).

Recent work in WSD has focused on the construction of word experts (Hoste *et al.*, 2002), where the set of features that works best is chosen from held-out data for each word. In this experiment, we can see that there are big differences for some words depending on the feature set we use (e.g.: *know* in DSO). We studied the results of the 8 common words in both sets to see whether the results where consistent, and we could separate local-words and topical-words. Only one word (*church*) worked better with topical features in both corpora, and 2 (*fall* and *know*) worked better on local features. The fact that for the other 5 words different sets where preferred in these two corpora shows again that the differences in tagging make difficult to extract conclusions.

III.5.4 Learning curve

One question that we should address is the quantity of data needed to train our supervised systems. With that goal, we trained our system with increasing quantities of data to see whether the system kept learning or it reached a standstill. The learning curve would be the graph resulting from this data.

In a previous paper (Ng, 1997), it is estimated that about 1,000 occurrences per word should be tagged to train a high accuracy domain-independent system. Based on this reference, Ng estimated that an effort of 16 person-years would be required to tag enough examples for the most frequent words in English (he proposed to use the most frequent sense heuristic with the rest of the words, which would account for less than 10% of all the occurrences, and those would be the less polisemous). Also in the mentioned work, he studied the learning curves of different word sets with a high number of examples, using the DSO corpus. He showed that the system kept improving when more data was added, even for words with more than 1,300 examples.

In this section, we studied the learning curve for our experimental setting. We performed the experiment in Semcor and DSO, using the same set of words (set B). We retained increasing amounts of the examples available for training each word: 10% of all examples in the corpus, 20%, 40%, 60%, 80%, and 100%. Cross-validation was applied for testing the results, and the process worked as follows: we partitioned the whole training set in 10 parts, and for each cross-validation step we used one different part for testing. From the remaining data, we chose the corresponding percentage (10%, 20%, 40%, ...) randomly for training.

The learning curve in Semcor is shown in figure III.10 and the DSO curve in figure III.11. In the figures, the Y axis marks the disambiguation performance (given as the recall, to normalize between the precision and the coverage), and the X axis indicates the number of examples. The averaged curves for each part of speech (nouns and verbs), and the overall curve are given.

The noun curve in Semcor shows that there is not enough data for a regular behavior. There are around 125 examples in average per noun, and each 20% implies that only 25 examples are added to the training set, which do not seem to make a difference for the higher partitions. On the contrary, for verbs we see a steady increase of recall when we train with more data. The overall results also show a constantly ascendant curve.

For DSO, the system keeps learning with more data, but it seems that there is no difference from 80% to 100%, suggesting that the system may have reached its top. At 80%, it uses an average of 930 examples per noun, and 1370 per verb. The performance is 72% for nouns and 67% for verbs. As we have seen in section III.5.1, the results for nouns are better in Semcor than in DSO.

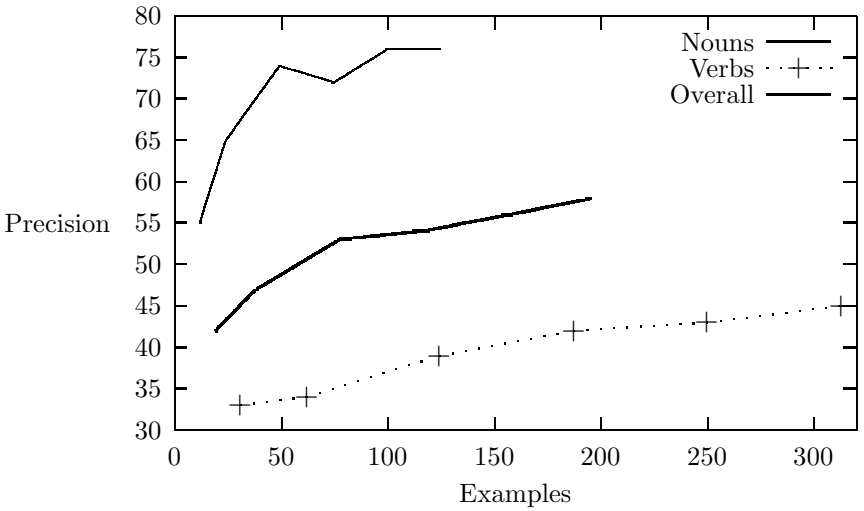

Figure III.10: Learning curve in the Semcor corpus.

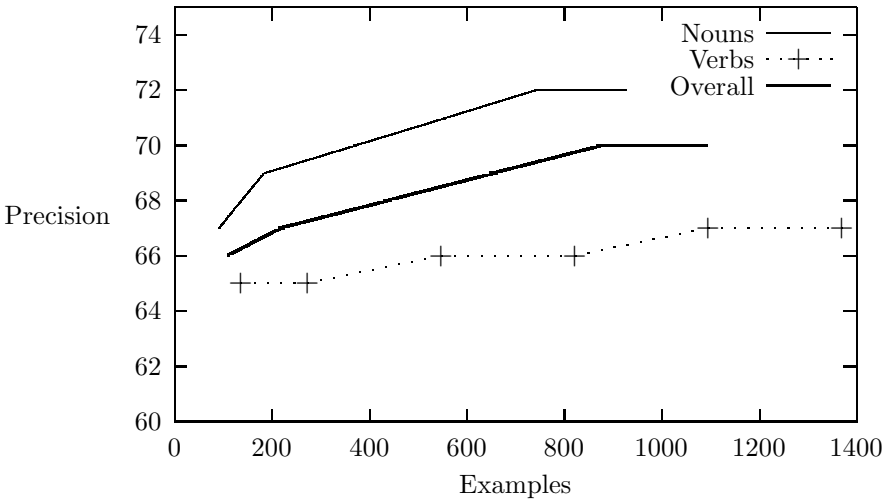

Figure III.11: Learning curve in the DSO corpus.

III.5.5 Noise in the data

In this section we want to evaluate the effect of noisy training data. The goal of this test is to analyze the performance of the system when the tagged data carries an expected amount of error. In a real setting, this would help

us to know what to expect when we obtain the training data automatically (with noise) instead of by hand.

The experiment works as follows: we introduced random incorrect tags in the examples, and created four new samples for training from each corpora (Semcor and DSO). Each sample had a fixed percentage of noise: 10% of the examples with random tags, 20%, 30% and 40%. We performed the experiment using the common set B of words. The results for each corpus are illustrated in figures III.12 and III.13. The graphics show the recall for each percentage of noise, from 0% to 40% . The averaged curves for nouns, verbs, and the overall curve are given.

In the figures, we can see that the performance on Semcor data drops instantly when 10% noise is introduced, and keeps decreasing constantly as we insert more noise in the data. In contrast, when the system is trained on DSO, it resists much better to noise, and only gets heavily affected when the percentage of noise reaches 40%. If we consider the curve of the nouns, we saw in section III.5.1 that for set B the Semcor data obtained better results, but it is enough to introduce 10% of noise in both corpora to eliminate this difference.

We can conclude that when we have few examples to train, as in Semcor, the noise affects the performance heavily, and it is necessary to use big amounts of data in order to minimize the damage.

III.5.6 Fine-grained vs. coarse-grained disambiguation

The choice of the sense inventory is a central discussion in WSD work. As we pointed out in the introduction chapter, for this research we chose to work with WordNet sense distinctions. This fixed inventory may be too fine-grained for many NLP tasks.

In this section we will measure the precision we can get using coarser senses than those defined in WordNet, but which can yet be useful for some applications. We will define this inventory using the WordNet architecture, which groups senses into semantic files. In these files, the synsets are grouped by part-of-speech and semantic similarity. Some examples of types of groups are the following: "nouns denoting acts or actions", "nouns denoting animals", etc. The complete list for nouns and verbs is given in table B.7 in the appendix. We can see how the grouping can be applied to the noun *age* in figure III.14.

For the experiment, we replaced the sense tags in Semcor and DSO data

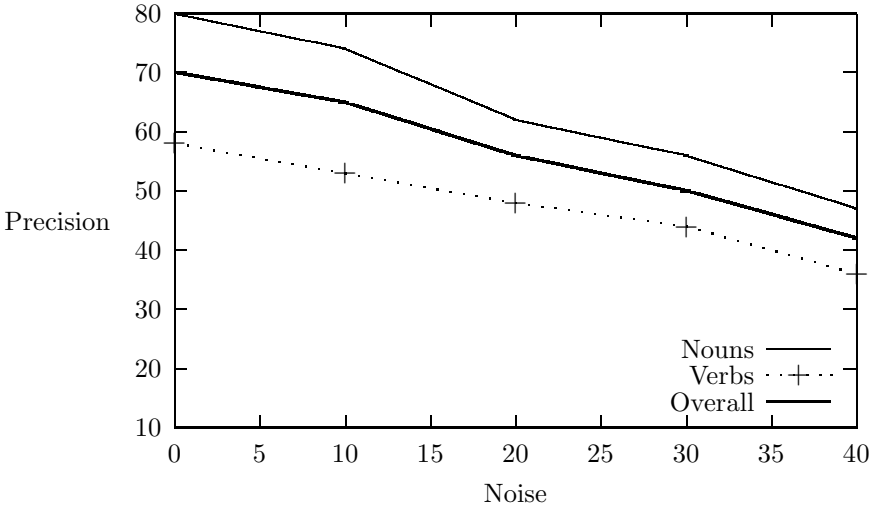

Figure III.12: Results with noise in the Semcor corpus.

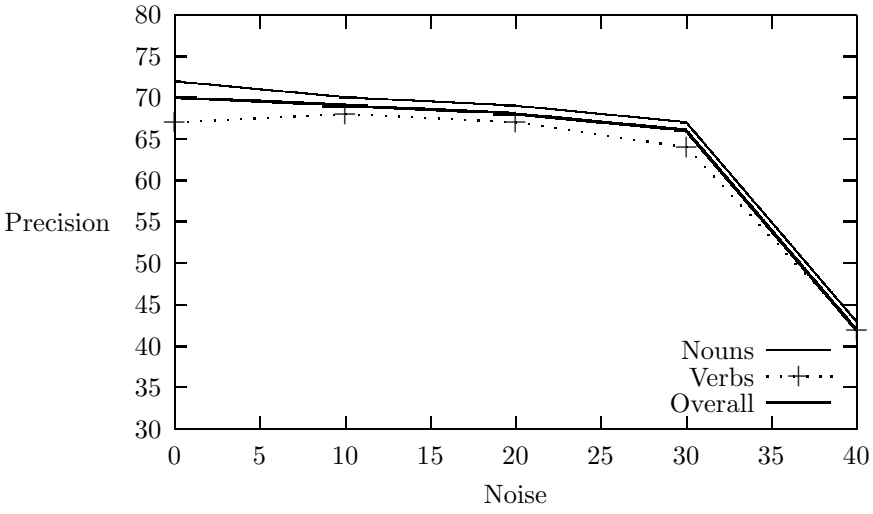

Figure III.13: Results with noise in the DSO corpus.

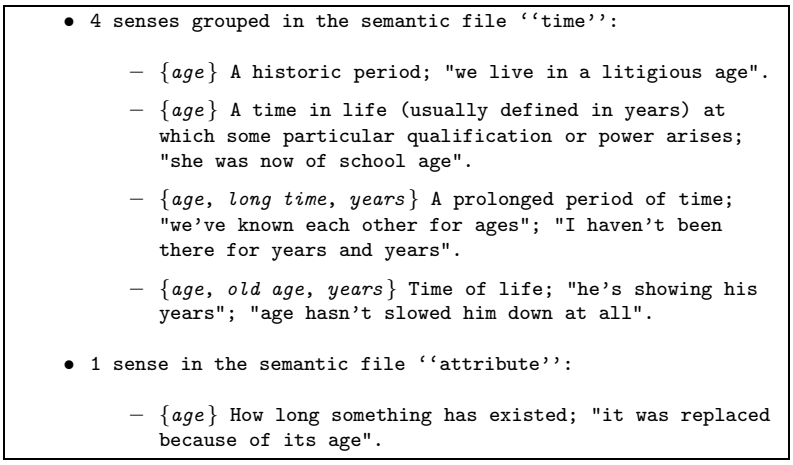

Figure III.14: Grouping for the noun *age*: 5 senses in WordNet 1.6 in 2 semantic files.

for their corresponding semantic file tags, and applied cross-validation as usual. Again, we used the common 8 word set (set B). Overall, the number of senses in average reduced from 17.25 senses to 6. The grouping of senses was stronger for verbs (from 29.3 to 6.3), while for nouns the granularity reduced from 10 to 5.8 . Table III.5 shows the results. The precision and coverage in Semcor and DSO is shown for synsets and semantic files. The results are averaged per PoS and overall.

The precision we obtain with semantic files is 83% in both corpora, with total coverage in DSO, and 98% coverage in Semcor. The results are significantly better than with synsets, but the amount of error (17%) is still important. For nouns, we can see that the improvement is small (1 precision point in Semcor, 4 in DSO), even when the average polisemy has been significantly reduced. We could attribute this low results to the scarcity of data, and the sense sparseness of the data points (the senses that appear in the corpus could belong frequently to different semantic files).

For verbs the results are much better. There is more training data, and the reduction in sense granularity is bigger, which enables the system to achieve good results. The best performance is obtained in DSO, where the 1,370 examples per word give a precision of 91%. Semcor provides 312 examples per word, and the precision reaches 87%. Again, we can see the importance of having enough training data in order to take profit of different techniques.

Words	PoS	# Synsets	# Semantic Fields	Semcor		DSO	
				Synset	SF	Synset	SF
age	N	5	2	76/100	75/100	73/100	74/100
church	N	3	3	69/100	69/100	71/100	71/100
head	N	30	15	88/100	88/100	79/100	80/100
interest	N	7	5	62/97	67/99	62/100	72/100
member	N	5	4	91/100	91/100	79/100	79/100
fall	V	32	7	34/71	57/71	80/100	85/100
give	V	45	10	34/78	72/95	77/100	87/100
know	V	11	2	61/100	100/100	46/98	100/100
Avg. N		10	5.8	77/99	78/100	72/100	76/100
Avg. V		29.33	6.33	51/90	87/96	67/99	91/100
Overall		17.25	6	62/94	83/98	70/100	83/100

Table III.5: Precision and coverage disambiguating coarse senses in Semcor and DSO.

III.5.7 Expected performance for all words in a text

For this experiment, we wanted to go beyond the fixed word sets and estimate which performance could we expect disambiguating all the content words in a corpus. In order to do that, we disambiguated the content words in four Semcor files, and also the nouns and verbs tagged in DSO.

Starting with the Semcor experiment, we disambiguated all content words in 4 files previously used in another work (Agirre and Rigau, 1996). In that work, the files were randomly chosen to test their unsupervised method, conceptual density, and compare it to other unsupervised methods (Yarowsky, 1992; Sussna, 1993). Although the direct comparison with DLs is not possible, because the experiments are defined differently, the results can give a reference of the performance of the unsupervised methods. The target files belong to different genre/topic: *Press-reportage* (br-a01), *Press: Editorial* (br-b20), *Learned: Science* (br-j09), *Humour* (br-r05).

We implemented the experiment as follows: for each word in the target file, we used the rest of the files as training data. The rare polisemous words with no examples in the rest of Semcor were left out of the experiment. In table III.6, we present the averaged results for each file, and the average results of the four files. The precision and coverage values shown correspond to the polysemous words in the files. Along with the precision and coverage values, the table shows the average number of senses per word in WordNet; the number of testing examples; and the precision of the MFS and random baselines, as reference.

File	Avg. Senses	Examples	Random	MFS	DL (prec./cov.)
br-a01	6.60	792	26	63	68/95
br-b20	6.86	756	24	64	66/95
br-j09	6.04	723	24	64	69/95
br-r05	7.26	839	24	63	68/92
Average	6.71	777.5	25	63	68/94

Table III.6: Overall results disambiguating 4 files in Semcor. Baselines have full-coverage.

PoS	Avg. Senses	Testing examples	Random	MFS	DL (prec./cov.)
Adjs.	5.49	122.00	28	71	71/92
Advs.	3.76	48.50	34	72	80/97
Nouns	4.87	366.75	28	66	69/94
Verbs	10.73	240.25	16	54	61/95

Table III.7: Overall results disambiguating 4 files in Semcor, given per PoS. Baselines have full-coverage.

The results display a similar performance for all files: around 68% precision. The baselines are also in the same numbers for the different files. The facts that, on the one hand the results are similar for texts from different sources (journalistic, humor, science), and on the other hand words with varying degrees of ambiguity and frequency have comparable performance (as seen in section III.5.2), seems to confirm that the training data in Semcor can provide these results across all kinds of words and texts, except for highly skewed words, where we can expect better performance than average.

In table III.7 the results are grouped according to the PoS of the target word. We can see that in this case there are significant differences.

- Verbs are the most difficult to disambiguate (as we saw in table III.2), obtaining only 61% precision. They have the highest polisemy (almost 11 senses per testing example). However, the precision is significantly better than the MFS baseline (7% higher).

- Nouns are the more frequent type in the testing data, totaling 47% of the examples. The performance for nouns is just above the overall average (69% precision and 94% coverage). The coverage for nouns is low

in comparison with the baseline word-sets experiment (section III.5.1), in this case it is even lower than the coverage for verbs.

- Adjectives achieve a precision of 71% and a coverage of 92%. There is no improvement over MFS, and the results are far from the 82% precision reported previously for the 2 adjectives in word set A.

- Adverbs obtain the best results: 80% precision and full coverage (8% better recall than baseline). These results confirm the good performance seen in table III.2.

The other experiment we conducted in this section was to disambiguate all the nouns and verbs tagged in DSO (121 nouns, 70 verbs). The DSO experiment gives us the opportunity to compare the results with other supervised approaches. In fact, Escudero *et al.* (2000b) present their results disambiguating the DSO corpus by means of different ML algorithms, concretely:

- K-nn: K-nearest neighbors exemplar-based method.

- AB: A variant of Shapire and Singer's AdaBoost.MH.

- NB: Naive Bayes.

For the implementation, we used 10 fold cross-validation to apply the DLs and to measure the MFS baseline, as in (Escudero *et al.*, 2000b). Table III.8 reports the precision and coverage of our approach, and the precision reported for the baseline and the other ML methods (always with full coverage).

The table shows that DLs obtains similar results to AB in precision, and slightly better than K-nn and NB. The results serve to illustrate that DLs can achieve state-of-the-art performance, but only as a reference. Even if the training and test data is the same, there are differences in the experiments: the feature representation, the cross-validation procedure, and the evaluation (value of partial answers) can lead to differences. For example, the MFS baseline is computed differently. In (Escudero *et al.*, 2000b) the precision is 52.3%, while in our case in reaches 56%.

Moreover, Hoste *et al.* (2002) show that the performance of different experiments can suffer large fluctuations depending on the following factors: the ML method and its parametrization, the kind of features used to represent the examples, and the interaction between the features and the parameters of

PoS	MFS	K-nn	NB	AB	DL
Nouns	59	69	68	71	**72**/99
Verbs	53	65	65	67	**68**/98
Overall	56	67	67	**70**	**70**/99

Table III.8: Overall results disambiguating DSO for different ML methods. Coverage is given for DLs only (other methods have full-coverage).

the algorithm. Recent works show that DLs perform worse when compared with more sophisticated methods, see (Yarowsky and Florian, 2002; Villarejo *et al.*, 2004), or the comparison between DL and AB in section IV.6. In the next section we will analyze the performance of DLs in a controlled framework to compare different approaches: the Senseval competition.

To conclude, in this section we have seen that we can reach a precision of 68-70%, for a coverage of 94%-99% tagging all the content words in a corpus. We have also seen that the results are similar for different types of texts and words, with some exceptions:

- Words with high bias are easier (cf. section III.5.2).
- The performance depends on the PoS of the words. Verbs tend to be the most difficult.

We have also introduced results of other approaches with the same corpora. We will study this further in the next section.

III.5.8 Comparison with other methods in a real setting: Senseval2

In this experiment, we evaluated our method in the Senseval-2 competition, which was reviewed in section II.8. This gave us the chance to compare the performance of our system with many other algorithms. The Senseval-2 framework presented different tasks in ten languages. The main tasks were the disambiguation of all the content words in a corpus (all-words task), and the disambiguation of selected words in different contexts (lexical-sample task). Normally, training data was provided for the lexical samples, but not for the all-words tasks. We present here our basic system for different tasks in Senseval-2; a different version including combination of algorithms, richer features, and smoothing was presented in Senseval-3. This last version and its performance is described in chapter V.

We participated in three tasks in Senseval-2: English lexical-sample, English all-words, and Basque lexical-sample. For the lexical-sample tasks, we used the training data provided by the organization; for the all-words task we used the Semcor corpus, which required a WordNet version mapping (Daude *et al.*, 2000). We presented a total of 5 systems, and in this section we will describe the two basic English experiments: English lexical sample and English all-words. We also submitted the results of systems based on feature selection (which will be described in section IV.7), and results for the Basque task (presented in section III.5.9). Our contribution to Senseval-2 is published in (Agirre and Martinez, 2001).

We had to adapt necessarily our setting for these experiments, because the corpus, the target words, and the sense inventory were different. We also decided to introduce a few changes in the basic feature set: we included bigrams and trigrams formed by lemmas, and we used lemmas (of content words only) instead of word-forms in the context windows. Besides, for the lexical-sample task, we used all the context provided by the organization instead of one single sentence. The Senseval-2 setting was described in section III.3.3.3, the details of our submission to the English tasks are given in figure III.15.

Before we analyze the results, we want to mention that we did not a complex pre-processing of the data (the features were extracted as described in section III.4.1). Certainly, the detection of multiwords would improve significantly the results (cf. section III.3.2). Respecting the all-words task, the unavailability of mapping for the adjectives, and the scarcity of data in Semcor affected strongly the coverage.

In order to compare the results for the lexical-sample task, we recall the table that we presented in section III.4.1, with our baseline system marked in bold. Thus, in table III.9, the performance of each of the 20 competing systems is given. The results are sorted by recall, and correspond to the fine-grained scoring. Only the last versions of resubmitted systems (R) are included. The baseline systems provided by the organization, which are not in the table, achieved the following recall: 51.2% (Lesk-corpus, cf. section II.6), 47.6% (MFS), and 14.1% (Random).

We can see that our system ranks 9th of 20 in precision and recall. With a recall of 56.4%, our simple implementation was not far from the more elaborate systems, and it was significantly better than the best baseline.

In the all-words task we obtained almost the same precision as in the lexical-sample task: 57.2%, but the coverage was limited to nouns and verbs

- Feature set
  - bigrams of word-forms, lemmas, and PoS.
  - trigrams of word-forms, lemmas, and PoS.
  - Bag of lemmas of the content words:
    - $\pm$ 4 word window around the target.
    - the whole context, usually the 2 preceding and 2 succeeding sentences (lexical-sample task).
    - the sentence (all-words task).
- Sense inventory: WordNet 1.7 pre-release, specially constructed for Senseval-2.
- Sense mapping
  - We performed an automatic mapping between the senses in Semcor (tagged with WordNet 1.6) and WordNet 1.7 pre-release .
  - Only nouns and verbs were mapped, due to time constraints.
- Lexical-sample experiment
  - 73 words (29 nouns, 29 verbs, and 15 adjectives).
  - Source corpus: BNC and WSJ.
  - 8611 tagged instances for training (approx. 118 per word).
  - 4328 instances for testing.
- All-words experiment
  - Testing: 5,000 words of running text.
  - Source corpus: WSJ articles from different domains of the Penn TreeBank II.
  - Semcor for training (via mapping).

Figure III.15: Setting for the Senseval-2 submissions (English tasks).

with training examples in Semcor, and reached only 51% of the target words. Besides, even if multiwords were marked, they were not properly identified for our experiments, and this was another source of error. Our system ranked 14th out of 21 in recall, and 7th out of 21 in precision. Table III.10 shows the results for the 21 systems[2], in this case there is no distinction between

[2]The system *CL Research-DIMAP* is assigned more than 100% coverage in the official results, due to some mistake.

Position	Precision	Recall	Coverage	System
1	64.2	64.2	100.0	JHU (R)
2	63.8	63.8	100.0	SMUls
3	62.9	62.9	100.0	KUNLP
4	61.7	61.7	100.0	Stanford - CS224N
5	61.3	61.3	100.0	Sinequa-LIA - SCT
6	59.4	59.4	100.0	TALP
7	57.1	57.1	100.0	Duluth 3
8	56.8	56.8	99.9	UMD - SST
9	**57.3**	**56.4**	**98.3**	**BCU - ehu-dlist-all**
10	55.4	55.4	100.0	Duluth 5
11	55.0	55.0	100.0	Duluth C
12	54.2	54.2	100.0	Duluth 4
13	53.9	53.9	100.0	Duluth 2
14	53.4	53.4	100.0	Duluth 1
15	52.3	52.3	100.0	Duluth A
16	50.8	50.8	99.9	Duluth B
17	49.8	49.8	99.9	UNED - LS-T
18	42.1	41.1	97.7	Alicante
19	66.5	24.9	37.4	IRST
20	82.9	23.3	28.0	BCU - ehu-dlist-best

Table III.9: Supervised systems in the Senseval-2 English lexical-sample task sorted by recall (version 1.5, published 28 Sep. 2001). Fine-grained scoring. R: resubmitted system. Our basic system (BCU - ehu-dlist-all) given in bold.

supervised and unsupervised methods. The format of the table is the same that we described for the lexical-sample task, and the version of the results is also 1.5 .

The organization provided a MFS baseline, which assumed perfect lemmatization, and did not attempt to find multiwords. The precision and recall of this baseline was 57%, which was very difficult to beat (only the three best systems achieved better recall). As we have seen, our system obtained comparable precision, but much lower coverage.

III.5.9 Evaluation on Basque in Senseval2

For the last experiment on this chapter, we tested the DL method on another language: Basque. Some characteristics, and the extraction of features for Basque are described in section III.4. As we did for English, a more sophisticated system was presented in Senseval-3, which is described in chapter V.

Three different teams took part in the Senseval-2 lexical sample task: Johns Hopkins University (JHU), Basque Country University (BCU) and University of Maryland (UMD). The third team submitted the results later,

Position	Precision	Recall	Coverage	System
1	69.0	69.0	100.0	SMUaw
2	63.6	63.6	100.0	CNTS-Antwerp
3	61.8	61.8	100.0	Sinequa-LIA - HMM
4	57.5	56.9	98.9	UNED - AW-U2
5	55.6	55.0	98.9	UNED - AW-U
6	47.5	45.4	95.5	UCLA - gchao2
7	47.4	45.3	95.5	UCLA - gchao3
8	41.6	45.1	108.5	CL Research - DIMAP
9	50.0	44.9	89.7	UCLA - gchao
10	36.0	36.0	99.9	Universiti Sains Malaysia 2
11	74.8	35.7	47.7	IRST
12	34.5	33.8	97.8	Universiti Sains Malaysia 1
13	33.6	33.6	99.9	Universiti Sains Malaysia 3
14	**57.2**	**29.1**	**50.7**	**BCU - ehu-dlist-all**
15	44.0	20.0	45.3	Sheffield
16	56.6	16.9	29.8	Sussex - sel-ospd
17	54.5	16.9	31.0	Sussex - sel-ospd-ana
18	59.8	14.0	23.3	Sussex - sel
19	32.8	03.8	11.6	IIT 2
20	29.4	03.4	11.6	IIT 3
21	28.7	03.3	11.6	IIT 1

Table III.10: Supervised systems in the Senseval-2 English all-words task sorted by recall (version 1.5, published 28 Sep. 2001). Fine-grained scoring. Our basic system (BCU - ehu-dlist-all) is given in bold.

Prec.	Recall	Attempted	System
75.7	75.7	100	JHU
73.2	**73.2**	**100**	**BCU-ehu-dlist-all**
70.3	70.3	100	UMD
64.8	64.8	100	MFS

Table III.11: Results in Senseval-2 in the lexical-sample Basque task.

out of the Senseval competition. The results for the fine-grained scoring are shown in table III.11, including the MFS baseline. Assuming full coverage, JHU attains the best performance. Our system obtained 73.2% precision for 100% coverage. The system improved in almost 9 points the precision of the MFS baseline, but was two points below the best system (JHU- Johns Hopkins University). We have to notice that the JHU system won the lexical sample task both for Basque and for English; and while the difference in recall with our system was only 2% for Basque, it reached 8% for English. We think that the reason for this is that our feature set for Basque is better, although our ML algorithm is worse.

III.6 Conclusions

Throughout this chapter, we have worked with an algorithm based on DLs and basic features in different experiments. Our goals in this analysis were to measure the performance we could achieve with this basic system (in order to compare it to the approaches in the next chapters), and also to study different aspects of the WSD problem on this setting.

We have seen that the Semcor corpus provides enough data to perform some basic general disambiguation, at 68% precision on any general running text. The performance on different words is surprisingly similar, as ambiguity and number of examples are balanced in this corpus. The main differences are given by the PoS of the target words: the verbs present the highest polisemy and lowest precision (11 senses in average, 61% precision), as it is usually the case.

The DSO corpus provides large amounts of data for specific words, allowing for improved precision. It is nevertheless unable to overcome the 70% barrier, and as we have mentioned in section III.5.1, the results for nouns are better in Semcor, due probably to tagging disagreements. Other works in the literature that rely on DSO have shown similar performance with ML algorithms like AB and NB (Escudero *et al.*, 2000b).

However, when applied to the Senseval-2 dataset, the system presents much lower performance, with a precision of 57% for the lexical-sample and all-words tasks (the recall was slightly lower in the lexical-sample, and much lower for the all-words). There are different reasons for these disappointing results. Focusing on the lexical-sample, we have to take into account that the best system only scored 64.4% recall, significantly lower than the 70% figures in Semcor and DSO. This indicates the difficulty of the word-set, where participating systems scored specially low with the verbs. Another factor to explain the low performance of our system was the importance of pre-processing the examples to detect multiword senses. Most of the top-performing systems included such a pre-process, and this affected the results[3]. Finally, our system was not optimized for performance, its goal was to be a baseline system for reference when applying different improvements (feature types, training example set, or combination with other algorithms). We will come back to these issues at the end of these conclusions.

[3]This factor motivated the construction of the multiword detection tool presented in section III.3.2

Following with the analysis of the Senseval-2 results, an early conclusion would be to blame the DL method on the low performance in front of more sophisticated ML methods. But as the work in (Hoste *et al.*, 2002) shows, the performance of these kind of systems is affected by three factors: the learning algorithm and the parameter setting, the feature set, and the interaction between them. As a reference, the DL algorithm was used in the ensemble presented by the winning team (JHU, cf. section II.6), and achieved 63% recall. The main differences with our system were the pre-processing, the inclusion of syntactic dependencies as features, and the weighting of feature types.

Up to now, in the conclusions we have addressed the performance we can expect for different tasks and corpora, including the Senseval-2 evaluation. Regarding the questions we posed at the beginning of the chapter, these are the main conclusions of our experiments, and how they affect the WSD system:

1. **Word types: relation between polisemy/bias/frequency and disambiguation performance**

   The highest results can be expected for words with a dominating word sense, but the difference to the MFS baseline is lower. Words with high polisemy tend to be the most frequent, which makes the polisemy and frequency factors balance each other. Therefore, in order to know previously which are the difficult words, we would require information about the frequency distribution of the senses, which is difficult to obtain.

2. **Feature types: relation between word types and basic feature types**

   Local features vs topical features. In our experiments the behavior was different depending on the corpus.

   - Semcor: topical features were better for nouns, but not for other categories. These results are consisted with the work by Leacock *et al.* (1998). Taking the results overall, local features performed better, and the recall for the whole set of features was similar to using only the local set (but with higher coverage for the whole set).

- DSO: the local features achieved better performance than the topical set for all categories. This could be due to the much higher number of examples in DSO. The best recall was obtained using the whole feature set, as topical features help to improve coverage.

It is important to note that single words exhibit different behavior, suggesting that the best policy could be the construction of word-experts with specific feature sets (Hoste *et al.*, 2002).

3. **How much data is needed? Learning curve.**

   The learning curve shows that Semcor has too few examples for these experiments. Specially nouns do not have enough data to see a regular behavior. For verbs we see a steady increase of recall when we train with more data. The overall results also show an ascendant curve. Therefore, more data would help to improve the WSD system.

   In DSO, the system keeps learning with more data, but it stabilizes with 80% of all the available data, which indicates that for this kind of system we have reached the roof. At that point, it uses an average of 930 examples per noun, and 1,370 per verb. The recall is 72% for nouns and 67% for verbs.

4. **How much noise can the data accept?**

   About this factor, we can conclude that when we have few examples to train, as in Semcor, the noise affects the performance heavily, and it is necessary to use bigger amounts of data in order to minimize the damage.

5. **Fine-grained vs. coarse-grained disambiguation.**

   The precision we obtain with semantic files is 83%, both in DSO and Semcor; but with slightly lower coverage in Semcor. The improvement is specially noticeable for verbs, where the reduction of sense granularity allows to reach 91% recall in DSO. An open issue is to find applications where coarse disambiguation would help.

6. **Expected performance for all words in a text.**

Our experiments in the Semcor and DSO corpora have illustrated that we can expect a recall of around 68%-70% when working on this kind of setting with the WordNet sense inventory.

7. **Comparison with other methods in a real setting: Senseval-2.**

   The results in this setting have been significantly lower (57% precision) due to problems related to the specific setting, as was explained in the beginning of this section.

8. **Study performance for another language, less studied and with less resources: Basque.**

   Our main conclusion regarding our work for Basque was that more work was needed on the feature set. Our aim was to imitate the expressiveness of the well-studied features for English WSD, and we introduced several different feature types with that goal. A better study of the contribution of single features would be desirable. In any case, the results in the Senseval-2 task are encouraging, with our system only 2% below the winning JHU system (while the difference was 8% between these systems for English), which would indicate that our feature set represented better the context than the JHU set, although their ML method was clearly better.

After the experiments with our baseline system, we are now able to start studying the main hypotheses of this dissertation: the contribution to the WSD problem of linguistically motivated feature representations, and the automatic acquisition of examples to alleviate the knowledge acquisition bottleneck.

Regarding the feature set, it is clear that the integration of diverse and informative features is necessary to move towards the solution of the problem. Our results in the Senseval-2 setting in comparison with other systems suggest that a richer feature set (including at least syntactic dependencies) should improve the performance of the system. In **chapter IV**, we will introduce different feature types based on syntactic dependencies, semantic tags, and selectional preferences in order to measure their contribution to disambiguation. We have also seen in our study of local/topical features that different words benefit from different feature types, and we will explore the possibility of choosing a different feature-set per word in a WSD system based on a trade-off between precision and coverage.

For the issue of the knowledge acquisition bottleneck, our experiments on the learning curves show that more data would help to improve the WSD systems. Moreover, the experiments on the effect of noise illustrate that the more examples we have, the smaller is the loss of performance in the presence of noise. Therefore, if we could obtain a big corpora with some noise on it (as it would happen with automatic means), it could be useful for WSD and could alleviate the hand-tagging effort. This issue is explored in-depth in **chapter VI**.

Finally, apart from the feature types and number of examples, we have seen in this chapter that our basic system should include other characteristics to be robust enough. For an enhanced version of our system to be tested in the next Senseval, we will include a multiword pre-processing tool (cf. section III.3.2), and we will explore other ML algorithms and smoothing techniques for better estimations of the training data (factors covered in **chapter V**).

IV. CHAPTER

New feature types: syntactic and semantic knowledge

IV.1 Introduction

In the previous chapter we mentioned that one of the limitations of our system was the representation of the context by means of simple features. As we noticed in the introduction chapter, the design of the feature-set is crucial when building a supervised WSD system. The features have to be generic enough to be applied in a variety of cases, and yet they should reflect the relevant information of the context at hand.

We will illustrate the importance of informative features by means of an example. Let us recall one of the questions that we introduced in chapter I:

Can you translate the whole document into Basque?

Let us assume that we want to disambiguate the verb *translate* in the sentence. For simplicity of the exposition, the goal will be to discriminate between the first two senses in WordNet 2.0, defined as follows:

1. translate, interpret, render – (restate (words) from one language into another language).

2. translate, transform – (change from one form or medium into another).

There are interesting features in the context that could be extracted by a dependency parser for the target word *translate*:

- Subject: *you*
- Head Object: *document*
- Head of Prepositional Phrase (into): *Basque*

In the basic feature set seen in chapter III, this information is handled by different features, which would represent the words *document* and *Basque* in the general "bag-of-words" feature; and the word *you* simply as a left collocate.

Now, let us suppose that we have in our training set the sentence *"I translated a book from Italian to Basque"*. If we rely on the basic feature set, the only feature that would match the test example is the general "bag-of-words" feature, instantiated with the word *Basque*. However, this feature type can match also several irrelevant words. On the other hand, using parsing information, the feature type "Head of Prepositional Phrase (into)" would match for *Basque*. This is a more discriminative feature type, and could help to discard noise.

Moreover, this kind of linguistic feature could be used to generalize further (with the WordNet hierarchy, for instance), and build selectional preferences, as in (Resnik, 1992, 1997). Following with the example, a training sentence like *"I translated a book into Spanish"* would be useful if the feature type "Head of Prepositional Phrase (into)" would be able to allow matching of classes that are below a superclass like "Languages", thus relating *Basque* and *Spanish*.

Traditionally, the WSD systems have relied on basic feature sets to learn their models. Only in recent years this picture has changed, with the advent of "off the shelf" parsing tools and other resources that can provide rich features, like the domain information from WordNet Domains (Magnini and Cavagliá, 2000). The use of these tools to extract features have been noticeable in the systems participating in Senseval, specially in the last edition.

When richer information is applied, normally the different feature sets are integrated together, and no study of the performance of different features is done. However, there is interesting work on the contribution of different feature types (including syntactic dependencies like the ones we will study on this chapter) in the works by Yarowsky and Florian (2002) and Lee and

Ng (2002). We will describe their conclusions on section IV.2. For our work, we think that it is important to measure the contribution of each knowledge source separately, and that is one of the goals of this chapter. This study would allow us to construct a feature-set in a principled way, and to avoid redundant or noisy features in our setting.

Another important aspect of disambiguation is that different words exhibit different behavior; as we have seen in the previous chapter, some words are better disambiguated relying on the local context while others take more profit from "bag-of-word" type features. Another example comes from the literature, in the work by Gliozzo *et al.* (2004) we can see that their method based on domains works well for some words (domain-words), and obtains very low performance with others. The concept of word-experts (systems tailored for each different target word) is getting strong in WSD research in recent years (Decadt *et al.*, 2004). Thus, as it is difficult to know which knowledge source will be useful for a word in a context, it would be interesting to explore as many sources as possible before we shape our word-expert. In this chapter, we have also tested the selection of features per word, starting from a big set of basic and syntactic features.

The feature types we are going to study consist on a broad range of syntactic features, semantic features extracted using the WordNet hierarchy, and selectional preferences learned from an all-words sense-tagged corpus (Semcor). The necessary syntactic knowledge will be extracted using the Minipar parser (Lin, 1998b), which we chose after comparison with some other available for research. We will use the dependency trees to implement experiments using the Semcor and DSO corpora, and compare the new setting to the basic feature set described in chapter III. We will also apply those features to the Senseval-2 setting, to be able to compare our results with other systems. Our last experiment on syntactic features will consist on algorithms that perform precision/coverage trade-off to obtain systems that can answer with high precision to part of the test instances. One of these algorithms will rely on selection of features per word; an approach that could also be useful to retrieve sense-tagged examples automatically in a fashion similar to the method in chapter VI.

Regarding semantic features, for our first experiments we will extract them from the context using the disambiguated corpus Semcor and the relations in the WordNet hierarchy. We defined features based on the synsets surrounding the target word, the hypernyms of these synsets (at different levels), and also their semantic files.

Finally, we will learn selectional preferences for classes of verbs using the syntactic dependencies in the Semcor corpus. We will rely on the WordNet hierarchy to assign weights to the relations between synsets, and we will apply the learned model to disambiguate the testing examples.

This chapter is organized as follows. In the next section we will describe works from the literature that are in line with the aims of the chapter. Section IV.3 will outline the setting we will apply in the different experiments. Section IV.4 will introduce the set of syntactic features that we will acquire from the dependency parser. The next three sections (IV.5, IV.6, and IV.7) will be devoted respectively to experiments performed with syntactic features in Semcor and DSO; to experiments in the Senseval-2 setting; and to the precision/coverage trade-off experiments. In the subsequent section, IV.8, the semantic feature set will be introduced, and the evaluation of the effect of those features will be covered in section IV.9. The focus of the next two sections (IV.10, and IV.11) will be selectional preference learning and the corresponding evaluation. Finally, the conclusions of the chapter will be summarized in section IV.12.

IV.2 Related work

The importance of integrating richer feature sets in WSD models is now reflected in the growing number of systems that apply them in some way. As we will see, the Senseval competitions and the recent literature offers many examples of this trend. We will present some of them according to the different knowledge sources they use: syntactic information (dependency relations), semantic features (sense tags, or other semantic tags from the context), and selectional preferences. We will perform experiments separately for them in this chapter, and we will try to measure their contribution to WSD performance.

IV.2.1 Syntactic features

The feature types that are being most widely applied recently are syntactic dependencies. The availability of "off the shelf" parsing tools, and some empirical evidence of their contribution (Yarowsky and Florian, 2002; Lee and Ng, 2002), have made them interesting for WSD research. In (Lee and Ng, 2002), they apply the statistical parser from (Charniak, 2000), and extract

a small set of features based on dependencies. They define different features depending on the target PoS, and apply 4 different ML methods in a battery of experiments. They report the best results until that day both on the Senseval-1 and the Senseval-2 English lexical-sample datasets. Syntactic features contribute significantly to the overall performance.

In (Yarowsky and Florian, 2002) a complete survey of parameter spaces is carried out, including syntactic features extracted by means of heuristic patterns and regular expressions over the PoS tags around the target word. They describe an ensemble of ML methods that competed for different languages in Senseval-2; we already introduced this system in section II.7. The main conclusions of their study are that the feature space has significantly greater impact than the algorithm choice, and that the combination of different algorithms helps significantly to WSD. Overall, the best results are obtained combining different ML methods, and using the whole feature set; but they notice that the syntactic features contribute less than local and topical features. They argue that the reason for this could be the higher sparseness of these features, and also the noise introduced in the detection of features. They also show that syntactic features help more in the disambiguation of verbs, and when applied with discriminative methods like DLs or Transformation Based Learning (TBL).

In the Senseval competitions for English, the number of systems using syntactic features has been growing. In Senseval-1 (cf. section II.6) only the winning system (JHU) applied these features among the top-performing systems. In Senseval-2 (cf. section II.7), again the winning system from JHU (described above) relied on syntactic features; and we can also mention (Tugwell and Kilgarriff, 2001), which obtains a grammatical relations database from the corpus, using finite-state techniques over PoS tags. This database is used to construct semi-automatically clues for disambiguation. The other systems did not apply these features in the English tasks, although one used dependencies to learn selectional preferences, as we will see below. In the 3rd edition of Senseval (cf. section II.8) many of the top ranked systems included syntactic dependencies in their feature sets. However, in the lexical-sample task, the two best systems did not have time to include them, but they mention that they would like to try them on the future. In the all-words task, the best performing systems relied on this type of knowledge, as separate features (GAMBL), or with semantic generalizations (SenseLearner), both these systems were described in detail in section II.8.

IV.2.2 Semantic features and selectional preferences

Another way to exploit richer information is to generalize from the words in the context using different techniques (generally with the aid of resources like the WordNet ontology). The process can benefit from the syntactic dependencies seen in the context and construct what is called "selectional preferences" of the target word. There are many approaches relying on this technique in the literature. Resnik (1992, 1997) defines an information-theoretic measure of the association between a verb and nominal WordNet classes: selectional association. He uses verb-argument pairs from the BC. Evaluation is performed applying intuition and WSD. The model we will introduce in section IV.10 follows in part from his formalization.

Abe and Li (1996) follow a similar approach, but they employ a different information-theoretic measure (the minimum description length principle) to select the set of concepts in a hierarchy that generalize best the selectional preferences for a verb. They call their model Tree Cut Model (TCM). The argument pairs are extracted from the WSJ corpus, and evaluation is performed using intuition and PP-attachment resolution.

In (Stetina *et al.*, 1998), they extract $[word-argument-word]$ triples for all possible combinations, and use a measure of "relational probability" based on frequency and similarity. They provide an algorithm to disambiguate all words in a sentence. It is directly applied to WSD with good results.

In (Stevenson and Wilks, 1999), selectional restrictions based on LDOCE semantic classes are applied in a "partial sense tagger" that is included in a combined system. They extract syntactic dependencies using a specially constructed shallow parser, and the sense-tagger only keeps the senses that do not break any constraint for the expected semantic classes of the arguments. The classes of LDOCE are organized hierarchically, therefore, the constraint is kept if the semantic category is at the same level or lower in the hierarchy.

Regarding the Senseval competitions, we described the system *LIA--Sinequa* in section II.6 as one of the best performing in Senseval-1. Their system trained Binary Decision Trees on a feature set that included WordNet semantic classes in fixed positions around the target word. In Senseval-2, the work by McCarthy *et al.* (2001) is an extension of the TCM model described above. In this case, the TCMs are acquired for verb classes instead of verb forms. They apply Bayes rule to obtain probability estimates for verb classes conditioned on co-occurring noun classes. They use the subject and object relations between argument heads. The main problem of this all-words sys-

tem was the low coverage, and they alleviate it relying on the "one sense per discourse" constraint, and in anaphora resolution. Finally, in Senseval-3 we can find "SenseLearner", already described in section II.8, which also applies semantic generalizations in one of its two modules. This system ranked second on the English all-words task.

IV.3 Experimental setting

The experiments performed in this chapter have followed two settings with their corresponding corpora, sense inventory, and word-sets (cf. section III.3.3): "Semcor&DSO" setting, and "Senseval2" setting. As ML methods, DL and AB have been applied. The new features will be described in section IV.4. The experiments are distributed as follows:

- Syntactic features: 2 settings:
  - Semcor&DSO setting: DL method.
  - Senseval2 setting: DL and AB methods.
- Semantic features: Semcor&DSO setting: DL method.
- Selectional preferences: Semcor&DSO setting.

IV.4 Syntactic features

In order to extract syntactic features from the tagged examples, we need a parser that meets the following requirements: free for research, able to provide the whole structure with named syntactic relations (in contrast to shallow parsers), positively evaluated on well-established corpora, domain independent, and fast enough.

We found three parsers that fulfilled all the requirements: Link Grammar (Sleator and Temperley, 1993), Minipar (Lin, 1998b) and RASP (Carroll and Briscoe, 2001). We installed the first two parsers, and performed a set of small experiments (John Carroll helped out running his own parser). Unfortunately, a comparative evaluation does not exist; therefore we performed a little comparative test, and all parsers achieved similar results. At this point

we chose Minipar mainly because it was fast, easy to install and the output could be easily processed to extract dependencies. The choice of the parser did not condition the design of the experiments, and the results should also be applicable to other parsers with similar performance.

From the output of the parser, we will extract different sets of features. First, we distinguish between direct relations (words linked directly in the parse tree) and indirect relations (words that are two or more dependencies apart in the syntax tree, e.g. heads of prepositional modifiers of a verb). An example can be seen in figure IV.1.

- *Henry* obj_word *listed*
- *listed* objI_word *Henry*
- *petition* mod_Prep_pcomp-n_N_word *listed*
- *listed* mod_Prep_pcomp-n_NI_word *petition*

Figure IV.1: Relations extracted for the verb *list* from the sentence *Henry was listed on the petition as the mayor's attorney. Obj_word*:verb-object relation. *Mod_Prep_pcomp-n_N_word*: relation of type "nominal head of a modifier prepositional phrase" between verb and noun. I: inverse relation.

We will describe the tuples that are extracted in the example. The direct relation "verb-object" is obtained between *listed* and *Henry* and the indirect relation "head of a modifier prepositional phrase" between *listed* and *petition*. For each relation we store also its inverse. The relations were coded according to the Minipar identifiers (see table IV.1). For instance, in the last relation in figure IV.1, *mod_Prep* indicates that *listed* has some prepositional phrase attached, *pcomp-n_N* indicates that *petition* is the head of the prepositional phrase, *I* indicates that it is an inverse relation, and *word* that the relation is between words (as opposed to relations between lemmas).

The most relevant relations are shown in table IV.1. For each relation this information is provided: the acronym of the relation, whether it is used as a direct relation or to construct indirect relations, a short description, some examples, and additional comments. The complete list of relations is given in table B.8 in the appendix.

Table IV.2 illustrates the way the different dependencies are related. We see that in order to extract the dependencies between words, we have to follow the relations that are given in Minipar. As the table shows, some dependencies are defined by 2 or 3 relations in Minipar. For each relation,

we show the PoS tags of the components and some examples. The PoS tags give information about the subcategorization of the words, and we will use them to build some features. Some dependencies represent strong relations (arguments), and are marked in bold. The complete tagset of Minipar is shown in figure IV.2.

We will classify the syntactic features as instantiated grammatical relations (IGR) and grammatical relations (GR).

IV.4.1 Instantiated Grammatical Relations (IGR)

IGRs are coded as {`wordsense relation value`} triples, where the value can be either the word form or the lemma. We also use the PoS information to construct the Minipar relations (e.g. `Mod Prep.`). The list of the relevant relations in Minipar (cf. table IV.1), and the connections in table IV.2 will be the base to select the relations that seem to have useful information, for a total of 38 features. Two examples for the target noun *church* are shown below. In the first example, a direct relation is extracted for the {`building`} sense (church#2 in WordNet 1.6), and in the second example an indirect relation for the {`group of Christians`} sense (church#1). The former relates directly the verb with its object, and the latter links the verb *surrender* to *church* (which is the head of a prepositional phrase) following two Minipar dependencies, namely *mod* (modifier) and *pcomp-n* (nominal head of PP).

- Example 1 : "...Anglican *churches* have been *demolished*..."
  {`Church#2 obj_lem demolish`}
- Example 2 : "...to whip men into a *surrender* to a particular *church*..."
  {`Church#1 mod_Prep_pcomp-n_N_lem surrender`}

IV.4.2 Grammatical relations (GR)

This kind of feature refers to the grammatical relation itself. In this case, we collect bigrams {`wordsense relation`} and also n-grams {`wordsense relation1 relation2 relation3 ...`}. The relations can refer to any argument, adjunct or modifier. N-grams are similar to verbal subcategorization frames, and at present, we have used them only for verbs. We want to note that Minipar provides simple subcategorization information in the PoS itself

Relation	D.	I.	Description	Examples	Comments
By-subj	X		Subj. with passive		
C		X	Clausal complement	... that John loves Mary that <-c- John loves Mary	
				I go there for + inf. clause go <-mod- (inf) <-c- for <-i- mainverb	
Cn		X	Nominalized clause	to issue is great be <-s inf <-cn inf <-i issue	Often wrong
Comp1	X		Complement (PP, inf/fin clause) of noun	... one of the boys one (N_P) <-comp1- of <-pcomp-n- boy	"boy in the garage" is MOD
				... grants to finance hospitals grants (N_C) <- c1- (inf) <-i- finance	
				... resolution which voted ... resolution (N_C) <-c1- (fin) <-i- voted	
Desc	X		Description	... make a man a child make <-desc- child	Occurs frequently
Fc	X		Finite complement	... said there is ... say <-fc- (fin) <-i- mainverb	
I		X	See c and fc, dep. between clause and main verb		
Mod	X		Modifier	...strikes increase as workers demand... increase <-mod as <-comp1 fin <-i demand	
				raises to cope with situation raise <-mod inf <-i cope <-mod with <-pcomp-n situation	
				... was already lost ... lost <-mod- already	
Obj	X		Object		
Pcomp-c	X		Clause of pp	in voting itself in <-pcomp-c vpsc <-i-votig	
Pcomp-n	X		Nominal head of pp	in the house in <-pcomp-n house	
Pnmod	X		Postnominal mod.	person <-pnmod missing	
Pred	X		Predicative (can be A or N)	John is beatuful (fin) <-i- is <-pred beautiful <-subj John	
Sc	X		Sentential complement	force John to do force <-sc-do	
Subj	X				
Vrel	X		Passive verb modifier of nouns	fund <-vrel- granted	When "pnmod", is tagged as adj. (often wrongly), here is tagged as verb

Table IV.1: The most relevant syntactic relations, with examples and comments. D: Direct relation. I: Indirect relation.

Source PoS	Dep.	PoS	Dep2.	PoS	Dep3.	PoS	Examples
V_N	**Obj**	N	-	-	-	-	
V_N	**Subj**	N CN	- NO	-	-	-	It does not link like s
A (?)	**Subj**	C	i	V			*it is possible to* *<-subj- be*
V_N	**S**	N CN	- cn	- C	- i	- V	*to buy is funny*
V_N	**By-subj**	Pr	pcomp-npcomp-c	N C	- i	- V	*made by J.* *made by cutting*
N *(no subcat)*	Mod	P A	pcomp-...				*end of doing,* *position of* *accepted practice*
A *(no subcat)*	Mod	P A	pcomp-n/-c				*essential for,* *fastidious in* *heavily traveled*
VBE	Mod	CPNA	i...				*is to enter* - - *is absolutely*
V *(no subcat)*	Mod	C P A	i... pcom...				*combine to investigate* *join after completing* *was aproved earlier*
C *(no subcat)*	Mod	PCAN	pcomp...				*On other matters, sbe. does ...*
N_A/_C/_P A_C/_P	**comp1**	A P C	pcomp-c/-n i	... V			(only N) *sth. close* *one of the day* *time to be*
V_N/V_A	**Desc**	A N					
N	Pnmod	A					*persons missing*
N	Vrel	V					*bonds issued by*
VBE	**Pred**	ANCP	i ... pco...				*there is a plan* *birs are to end* *is across ...*
V_C	**Fc**	C	i	V			subcat C: *have to face*
V_I	**Sc**	V					subcat I: *force sb to take*
V *no subcat*	Amod	A					*even know*

Table IV.2: Dependencies and their relations. The `PoS` columns indicate the Pos tag given by Minipar to the components of the relation; the `Dep.` (dependency) columns indicate the type of relation between the left and right elements. Examples are given in the last column. Arguments are marked in bold.

- Det: Determiners
- PreDet: Pre-determiners
- PostDet: Post-determiners
- NUM: Numbers
- C: Clauses
- I: Inflectional Phrases
- V: Verb and Verb Phrases
- N: Noun and Noun Phrases
- NN: Noun-Noun Modifiers
- P: Preposition and Preposition Phrases
- PpSpec: Specifiers of Preposition Phrases
- A: Adjective/Adverbs
- Have: Have Verb
- Aux: Auxiliary verbs. E.g.: should, will, does, ...
- Be: Different forms of be: is, am, were, be, ...
- COMP: Complementers
- VBE: Be as a linking verb. E.g.: I am hungry
- V_N: Verbs with one argument (the subject), i.e., intransitive verbs
- V_N_N: Verbs with two arguments, i.e., transitive verbs
- V_N_I: Verbs taking small clause as complement

Figure IV.2: List of PoS tags in Minipar.

(e.g.: `V_N_N` mark for a verb taking two arguments). We have defined 3 types of n-grams:

- Ngram1: The subcategorization information included in the PoS data given by Minipar, e.g. V_N_N.
- Ngram2: The subcategorization information in ngram1, filtered using the arguments that really occur in the sentence.
- Ngram3: All dependencies in the parse tree.

The three types have been explored in order to account for the argument/adjunct distinction, which Minipar does not always assign correctly. In the first case, Minipar's judgment is taken from the PoS. In the second case the PoS and the relations deemed as arguments are combined (adjuncts are hopefully filtered out, but some arguments might be also discarded). In the third case, all relations (including adjuncts and arguments) are considered.

In the following example, the *ngram1* feature indicates that the verb *fall* has two arguments (i.e. it is transitive), which is an error of Minipar, probably caused by a gap in the lexicon. The *ngram2* feature indicates simply that it has a subject and no object, and the *ngram3* feature denotes also the presence of the adverbial modifier *still*. *Ngram2* and *ngram3* try to repair possible gaps in Minipar's lexicon.

E.g.: `His mother was nudging him, but he was still` ***`falling`***`.`
`{Fall#1 ngram1 V_N_N}`
`{Fall#1 ngram2 subj}`
`{Fall#1 ngram3 amodstill_subj}`

IV.5 Syntactic features on Semcor and DSO

In this section we will describe the experiments that we performed on the Semcor and DSO corpora using the features defined in section IV.4. We targeted the experiments on setting "Semcor&DSO" (cf. section III.3.3.1); the 19-word set A was used when working on Semcor, and the 8-word set B with DSO. We applied DL, and exceptionally, for the experiments in this section pruning was not applied (cf. section II.4.2). The reason not to use pruning was that we could foresee that the coverage of the syntactic features separately would be low, and we expected good recall for the algorithm making decisions even with few data.

For our first experiment, we grouped the syntactic features in different sets, according to the description given in section IV.4 . For the IGR, we separated the relations obtained directly and indirectly; for the GR, we distinguished between direct and indirect bigrams, and we also separated the three types of n-grams described. There was a total of seven sets of syntactic features. We also applied the algorithm to the basic features described in

Feature set	Adj.		Adv.		Nouns		Verbs		Overall	
	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.
MFS	77	100	58	100	69	100	51	100	61	100
Basic feats.	82.5	100	69.9	100	79.3	100	**51.2**	100	**67.0**	100
IGR - direct	86.5	31.2	71.1	20.2	78.2	69.0	49.1	69.2	64.0	53.9
IGR - indirect	**100**	1.0	**100**	0.7	**90.9**	8.0	47.9	19.3	59.2	9.9
GR - bigr. - direct	79.3	89.2	56.5	70.8	70.5	92.8	43.8	87.3	58.7	85.5
GR - bigr. - indirect	81.9	5.4	81.7	8.1	62.3	45.3	44.5	51.2	53.7	34.7
GR - ngram1							45.3	99.7		
GR - ngram2							41.9	92.7		
GR - ngram3							47.2	66.4		

Table IV.3: Basic and Syntactic feature sets in Semcor. Precision and coverage of syntactic feature sets, basic feature set, and MFS baseline. Results per PoS and overall. Ngram features applied only for verbs. Best precision given in bold for each column.

section III.4.1, to know the performance we could achieve without pruning.

The results of this experiment, which was targeted to set A in Semcor, using 10-fold cross-validation, are shown in table IV.3. For each part-of-speech and overall, the precision and coverage of the seven syntactic feature sets, the basic feature set, and the MFS baseline are provided. The ngram features, which provide subcategorization information, were applied only for verbs. For each precision column, the best result is given in bold.

The syntactic feature sets exhibited different behavior:

- GR-bigram-direct was the only feature set that obtained acceptable coverage overall (85%), but its precision was lower than the basic feature set and the MFS baseline.

- The sets GR-ngram1 and GR-ngram2 obtained good coverage for verbs, but they also had lower precision than the baselines. We have to notice that the MFS baseline for verbs was as good as the DLs with the basic set of features, which made it difficult to beat.

- The IGR-direct feature set was better in overall precision than the MFS baseline, but for a coverage of 53%.

- The indirect feature sets obtained high precision, except for verbs, but could only be applied in a few cases.

Feature set	Adj.		Adv.		Nouns		Verbs		Overall	
	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.
MFS	77	100	58	100	69	100	51	100	61	100
Base Features	82.5	100	69.9	100	79.3	100	51.2	100	67.0	100
+ IGR-direct	**82.8**	100	**70.3**	100	**79.4**	100	51.1	100	**67.1**	100
+ IGR-indirect	82.5	100	69.9	100	79.2	100	**51.4**	100	67.0	100
+ GR-bigr-direct	82.7	100	69.9	100	79.1	100	51.2	100	66.9	100
+ GR-bigr-indirect	82.5	100	69.9	100	79.1	100	51.1	100	66.9	100
+ GR-ngram1							51.3	100		
+ GR-ngram2							51.2	100		
+ GR-ngram3							51.3	100		

Table IV.4: Performance in Semcor adding syntactic features to the basic set. Best precision per column given in bold.

From this experiment we have to conclude that when used separately, the syntactic features that we extracted present a lower performance than the basic feature set.

In our next experiment, we used the syntactic feature sets in combination with the basic set of features. We expected the new features to provide additional clues that would improve the performance of the basic setting. Thus, we repeated the previous experiment combining the basic set and the syntactic sets. The results are shown in table IV.4. As in the previous experiment, the seven new feature sets and the two baselines are provided per PoS and overall, and the best precision for each column is marked in bold.

We see clearly that there is no improvement over the results of the basic set. It seems that the syntactic features do not add new information for the DLs. Before we analyze these results in more detail, we repeated the experiment on the DSO corpus for the set B of words (cf. section III.3.1.1). We expected that this would help to improve the coverage of the syntactic features, because the number of examples per word is higher in this corpus. The results are illustrated in table IV.5. For this experiment, we did not separate direct and indirect relations, and we included all the syntactic features in a new set (GR + IGR). The precision and coverage of the syntactic set, and the combination of basic and syntactic sets is shown. The best precision per column is given in bold.

We can see that the coverage is still poor, but the precision is higher than in Semcor. The MFS baseline is easily beaten, the IGR features improve significantly the precision of the basic set overall (with lower coverage), and

Features	Nouns		Verbs		Overall	
	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.
MFS	56	100	61	100	59	100
Base Features	73.1	100	69.1	99.4	71.2	99.7
IGR	**76.2**	24.3	71.4	28.2	**73.7**	26.1
GR-bigram	68.4	35.0	**71.7**	26.8	69.8	31.0
GR-ngram1			46.3	40.6		
GR-ngram2			52.9	37.6		
GR-ngram3			54.5	28.2		
IGR + GR	71.3	35.9	69.3	34.8	70.3	35.4
Basic + IGR	73.2	100	69.5	99.4	71.4	99.7
Basic + GR-bigram	73.2	100	69.2	99.4	71.3	99.7
Basic + GR-ngram1			67.9	100		
Basic + GR-ngram2			67.9	100		
Basic + GR-ngram3			68.0	100		
Basic + IGR + GR	73.3	100	69.6	99.5	71.5	99.8

Table IV.5: Results for basic and syntactic feature sets in DSO.

the GR features improve the base results for verbs. The IGR features exhibit a better behavior with nouns, and the GR features with verbs. Combining all the features, there was a small improvement over the basic set overall (0.3% in precision, and 0.1% in coverage); and a bigger difference for verbs (0.5% in precision, 0.1% in coverage).

At this point, we analyzed the behavior of the different features separately, in order to know whether they can be useful for disambiguation. We applied the DL algorithm using only one feature each time, and we evaluated the precision and coverage (normally low) of each piece of evidence. We included in this experiment the basic features, for comparison. For a better analysis, we separated the results by PoS, and sorted the features following two criteria: precision and recall. The tables are too large to be included here, and can be seen in section B.4 in the appendix.

Concerning the precision of the features, we see that all the high-precision features are syntactic, even if they attain very low coverage. There are many features with 100% precision, but they are applied few times. For instance, the full-precision feature that attains highest coverage for nouns is `mod_Prep_pcomp-n_N_lem` ("lemma of the head of a modifier prepositional phrase"). For reference, we already presented an example with a similar relation for verbs in figure IV.1. This relation can be used only in 3.3% of the examples (32 out of 959). For verbs there are few full-precision features, and their coverage is reduced to a handful of examples. The best is the feature `descI` (description), which appears only 4 times (always with the verb *die*

in the same sense).

However, when we sort the features according to their recall, the syntactic features are outperformed by basic features. For nouns, only basic features achieve more than 25% in recall. The best performing features are context windows, followed by local features (bigrams and trigrams) formed with PoS, local features formed with word-forms, and finally syntactic features. The best syntactic features are formed with the prepositional complements of the nouns, with 22.5% recall.

For verbs the results are different. The recall is lower, as it usually happens, and the features can be ranked as follows (range of recall given between parentheses):

1. Context windows (45%-50%)
2. GR-ngram1 (45.2%)
3. Local features formed with PoS (39%-43.5%)
4. GR-ngram2 (38.8%)
5. Syntactic features (leaded by GR-ngram3, subject, ...) and other local features (0%-38.1%)

In this case, the syntactic features obtain better results. The GR-ngram sets obtain good recall, even better than some basic bigrams and trigrams. This indicates that some subcategorization information has been acquired. Other syntactic features that appear high in the table are those related to the subject of the target verb, but attain lower coverage. In the tables from the appendix (cf. section B.4), where the results for all the features are given, we can notice that many syntactic features do not appear in the corpus.

Finally, in order to understand the reasons of the small improvement when combining all the features, we focused on some words in the Semcor experiment, and analyzed the learned decision lists. These are the main conclusions:

1. Syntactic features usually have fewer occurrences in the training corpus than basic features, and they rank low in the decision lists. Even syntactic features with high frequency in training usually have basic features above them, which suggest that the information may be redundant.

2. In the case of words with a dominant sense, some syntactic features that appear frequently and do not carry much information (e.g. the presence of a determiner linked to a noun) can introduce noise and point strongly to the most frequent sense. This happens also with non-syntactic features, but in a less harming scale because they comprise a more reduced and controlled set.

3. The parser fails to detect many dependencies and commits some errors, which affects the precision and specially the coverage.

From this first analysis of the syntactic features, we can conclude that different ways have to be explored in order to take advantage of this source of information. These issues will be addressed in different sections of this chapter:

- Section IV.6 covers the performance of the syntactic features in the Senseval-2 setting. By means of these experiments we will see the performance of another ML algorithm (AB) that will learn better from redundant features, and also the quality of the relations extracted from the Senseval-2 lexical sample corpus.

- Sections IV.10 and IV.11 in this chapter, introduce selectional preferences learned from some of these syntactic relations.

- Feature selection is presented in section IV.7, together with another method to improve precision at the cost of coverage. Feature selection can be a way to discard noisy syntactic features. In related work, smoothing of features, described in chapter V, will provide better estimations from the training data.

IV.6 Syntactic features on the Senseval-2 corpus

For the following experiments, we will use the Senseval-2 lexical sample corpus, and evaluate the effect of the syntactic features. We described in section III.5.8 the basic experiments we performed for English in this competition. The experiments that we will show in this section were not included in our Senseval-2 submission due to time constraints.

We devised two experiments in order to measure the contribution of syntactic features in the Senseval-2 setting. We will use the training part of the

PoS	IGR			GR			GR + IGR			MFS
	Prec.	Cov.	F1	Prec.	Cov.	F1	Prec.	Cov.	F1	F1
A	**81.6**	21.8	29.2	70.1	65.4	55.4	70.7	**68.9**	**57.7**	59.0
N	**74.6**	36.0	38.5	65.4	57.6	47.8	67.6	**62.5**	**52.0**	57.1
V	**68.6**	32.2	33.4	67.3	41.2	39.2	66.3	**52.7**	**45.4**	40.3
Ov.	**72.9**	31.9	35.2	67.1	52.1	46.0	67.7	**59.5**	**50.4**	48.2

Table IV.6: Performance for different sets of syntactic features (IGR, GR, GR+IGR) applying DL on the Senseval-2 corpus. The figures in bold indicate the best precision, coverage, and F1 per each PoS and overall.

lexical-sample corpus for training, and the testing part for evaluation; we will apply the "Senseval2" setting (cf. section III.3.3.3), for all the words in the lexical-sample task, and the WordNet 1.7Pre sense-inventory. First we will measure the performance of IGR-type and GR-type relations using DLs. Next, we will evaluate the benefit of adding syntactic features to the basic feature set using DLs and AB.

Performance is measured as precision and coverage. We also consider F1 to compare the overall performance, because it gives the harmonic average between precision and recall (where recall is in this case precision times the coverage). F1 will help us to compare the results of the two ML algorithms.

For our first experiment, table IV.6 shows the precision, coverage and F1 figures for each of the syntactic feature sets as used by the DL algorithm (IGR, GR, GR+IGR). The figures in bold indicate the best precision, coverage, and F1 per each PoS and overall.

IGRs provide very good precision, but low coverage. The only exceptions are verbs, which get very similar precision for both kinds of syntactic relations. GRs obtain lower precision but higher coverage. The combination of both attains best F1, and is the feature set used in the next experiment. Note that the combination of syntactic features is able to outperform MFS overall, and for verbs the increase in F1 is 5.1%. This indicates that the features represent useful information.

As a reference, before we present the main experiment we will compare these results with the performance of the previous section with the DSO corpus. Even if the experiments are different (different word-sets and corpora), this will give us a better idea of the performance we can achieve with this kind of feature. The rows in table IV.7 indicate the three feature sets and the MFS baseline; the columns represent nouns and verbs in the DSO and

Feature set	DSO		Senseval-2	
	Nouns	Verbs	Nouns	Verbs
IGR	29.8	31.4	39.5	33.4
GR	35.5	30.3	47.8	39.3
IGR+GR	37.7	35.8	52.0	45.8
MFS	56	61	57.1	40.3

Table IV.7: F1 values for the syntactic feature sets (IGR, GR, IGR+GR) in Senseval-2 and DSO for nouns and verbs.

PoS	MFS	IGR + GR		Local		Basic		Basic + IGR + GR	
		DL	AB	DL	AB	DL	AB	DL	AB
A	59.0	57.7	**62.6**	66.3	**67.5**	65.3	**66.2**	65.4	**67.7**
N	57.1	52.0	**60.0**	63.6	**65.3**	63.2	**67.9**	63.3	**69.3***
V	40.3	45.4	**48.5**	**51.6**	50.1	51.0	**51.6**	51.2*	**53.9***
Ov.	48.2	50.4	**55.2**	**59.4**	59.3	58.5	**60.7**	58.7	**62.5***

Table IV.8: F1 results for different algorithms, feature sets, and PoS in Senseval-2. '*' indicates statistical significance (McNemar's test) over basic set.

Senseval-2 corpora.

Senseval results are higher than DSO results in all cases. However, the MFS baseline (which can serve as an indicator of the difficulty of the task: the higher the MFS value, the easier the disambiguation) is similar in both corpora for nouns, but much higher in DSO for verbs. The better results in Senseval-2 over the baseline indicate that the features are more reliable in the Senseval-2 setting.

For our next experiment, DLs and AB were used on syntactic features, local features, a combination of local+topical features (also called basic), and a combination of all features (basic+syntax) in turn. Table IV.8 shows the F1 figures for each algorithm, feature set and PoS. Regarding the contribution of syntactic features to the basic set, the last two columns in the table include the character '*' whenever the difference in precision over the basic feature set is significant according to McNemar's test (cf. section II.5.2).

AB is able to outperform DLs in all cases, except for local features. The characteristics of the methods can be seen in section II.4. Syntactic features get worse results than local features, but prove to be useful in the combination. Focusing on the contribution of syntactic features, we see that DLs profit from the additional syntactic features but the difference is only sta-

tistically significant for verbs. On the other hand, AB attains significant improvement (1.8% overall, 2.7% for verbs). The results (specially for AB) show that basic and syntactic features contain complementary information, and that they are useful for WSD.

IV.7 Syntactic features and high precision systems

A high-precision WSD system can be obtained at the cost of low coverage, preventing the system to return an answer in the lowest confidence cases. With this kind of approach, we can tag part of a raw corpus with high confidence, to be able to learn from sense-tagged occurrences of a word. This would be useful, for instance, to learn selectional preferences with methods like the ones described in section IV.8; or for WSD in an iterative way, as in Yarowsky (1995b).

Methods to identify good features can be applied to high precision systems. This approach would allow to identify features that work well for specific words, and discard noisy features. At this point, where the evidence suggests that the large set of syntactic features that we have introduced should be refined (see section IV.6), feature selection seems a reasonable path to explore. Thus, we have built one high-precision system based on **feature selection** with DLs, which consist on choosing a reduced feature set in cross-validation for each word.

Other two high-precision approaches have also been tried with DL and AB, following the method in (Dagan and Itai, 1994) that applies **thresholds** to the decisions of the algorithms. As in section IV.6, the experiments have been performed in the Senseval-2 lexical sample setting.

We will start describing the **feature selection method**. Ten-fold cross validation on the training data for each word was used to measure the precision of each feature in isolation. Thus, the DL algorithm would be used only on the features with precision exceeding a given threshold. This method has the advantage of being able to set the desired precision of the final system. For example, for the noun *art*, using both basic and syntactic features, and a threshold of 80% precision, the system chooses the following features in cross-validation:

- has_relat_mod_inI
- lex-mod_lem

Threshold	Basic + IGR + GR			Basic		
	Prec.	Cov.	F1	Prec.	Cov.	F1
0	59.4	97.9	58.7	59.3	97.5	58.5
50	59.7	97.9	59.1	58.4	97.5	57.7
60	66.1	83.5	**60.2**	66.3	79.5	**58.7**
65	68.9	75.9	59.5	69.2	71.5	57.7
70	72.1	63.7	56.1	73.3	56.9	53.2
80	79.6	48.2	51.8	80.4	40.1	46.0
85	83.7	35.7	44.0	83.8	27.4	36.0
90	86.7	26.4	36.2	88.9	15.4	23.7
95	86.5	19.7	28.5	88.1	8.9	14.4

Table IV.9: Feature selection method with DL on two feature sets: basic features, and extended set with syntactic features (IGR + GR). Micro-averaged performance for each threshold for the 73 words in the Senseval-2 lexical-sample. In bold, the best F1 for each feature-set.

- nn_lem
- nn_word
- trig_lem_0
- trig_wf_0

Therefore, these would be the only features taken into account to tag occurrences of *art* in testing. We have used the following precision-thresholds: 50, 60, 70, 80, 85, 90, and 95. The application of the same threshold to all words provokes that some words can have no features selected for a given threshold, and others do not have any feature filtered. More complex feature selection methods would overcome this problem, but as we will see, this simple approach is enough to provide answers with high confidence for a percentage of the testing data.

The results of the feature selection experiment using the basic feature set, and the one extended with syntactic features (IGR + GR) are given in table IV.9. For each threshold, the table shows the precision, coverage, and F1. The results correspond to the 73 words in the Senseval-2 lexical-sample task (micro-averaged). The best F1 for each feature-set is given in bold.

From the overall results, we can derive these conclusions:

- The syntactic features improve significantly the results. Specially for the high-precision thresholds, the algorithms profit clearly from syntactic features.

- The best F1 is reached with the syntactic features and the 60% threshold, improving the performance of the whole set. This shows that some filtering of syntactic features is required.

- A high precision of 86.7% can be achieved for 26.4% of the testing examples.

The second method is based on a **decision-threshold** (Dagan and Itai, 1994): the algorithm rejects decisions taken when the difference of the maximum likelihood among the competing senses is not big enough. For this purpose, a one-tailed confidence interval was created so we could state with confidence $1 - \alpha$ that the true value of the difference measure was bigger than a given threshold (named θ). As in (Dagan and Itai, 1994), we adjusted the measure to the amount of evidence, and applied a 60% confidence interval. For each feature f and sense i, the lower bound $\beta_\alpha(feat_f, sense_i)$ was calculated using the following formula:

$$\beta_\alpha(feat_f, sense_i) = \log(\frac{N_{fi}}{\sum_{j \neq i} N_{fj}}) - Z_{1-\alpha}\sqrt{\frac{1}{N_{fi}} - \frac{1}{\sum_{j \neq i} N_{fj}}}$$

Where N_{fi} denotes the frequency of feature f with sense i, and $Z_{1-\alpha}$ is the confidence coefficient from the normal distribution. Thus, for a new example to disambiguate, a feature f is discarded for a sense i when $\beta_\alpha(feat_f, sense_i) < \theta$. The values of θ range from 2 to 4. The results are shown in table IV.10.

In this experiment the contribution of the syntactic features is smaller than in table IV.9, although they help. The use of all the features allows for a precision of 93.7% and a coverage of 7.9%.

In the case of AB, there was no straightforward way to apply the feature selection method. The application of the decision-threshold did not yield satisfactory results, therefore we turned to using the support value returned for each decision that was made. We first applied a threshold directly on this support value, i.e. discarding decisions made with low support values. A second approximation, which is the one reported here, applies a threshold over the difference in the support for the winning sense and the second

θ	Base + IGR + GR			Base		
	Prec.	Cov.	F1	Prec.	Cov.	F1
2	63.2	85.4	**58.2**	61.9	84.4	**56.7**
2.5	70.7	64.1	55.2	69.2	62.6	53.3
3	79.7	39.0	44.7	78.7	37.7	43.1
3.5	88.4	20.5	30.1	88.5	19.0	28.3
4	93.7	7.9	13.7	93.5	7.1	12.4

Table IV.10: Decision threshold with DL on two feature sets: basic features, and extended set with syntactic features (IGR + GR). Micro-averaged performance for each θ for the 73 words in the Senseval-2 lexical-sample. In bold, the best F1 for each feature-set.

Basic + IGR + GR			Basic		
Prec.	Cov.	F1	Prec.	Cov.	F1
60.3	100	60.3	59.4	100	59.4
66.7	84.5	**61.1**	65.6	83.5	**59.7**
68.5	77.5	59.8	67.7	76.3	58.6
70.4	70.0	58.0	69.5	69.4	56.9
71.9	61.9	55.0	70.6	61.4	53.7
74.5	45.1	46.3	73.3	43.5	44.4
74.9	38.4	41.6	74.0	37.7	40.5
75.0	33.5	37.6	73.1	32.0	35.4
74.0	26.7	31.2	72.5	27.5	31.3
72.0	19.3	23.3	71.0	18.6	22.3
92.1	0.9	1.6	0.9	89.5	0.9
100	0.8	1.6	0.8	96.7	0.8

Table IV.11: Application of AB with decision thresholds. Performance on intermediate points for basic and extended features. Micro-averaged performance for the 73 words in the Senseval-2 lexical-sample. In bold, the best F1 for each feature-set.

winning sense. Still, further work is needed in order to investigate how AB could abstain in the less confident cases.

The results for some representative points are given in table IV.11. There, we can see that the F1 value improves significantly for syntactic features, reducing a little the coverage. However, the system is not able to reach high percentage values, except for a handful of cases. As we said, other means should be explored to adapt AB to this task.

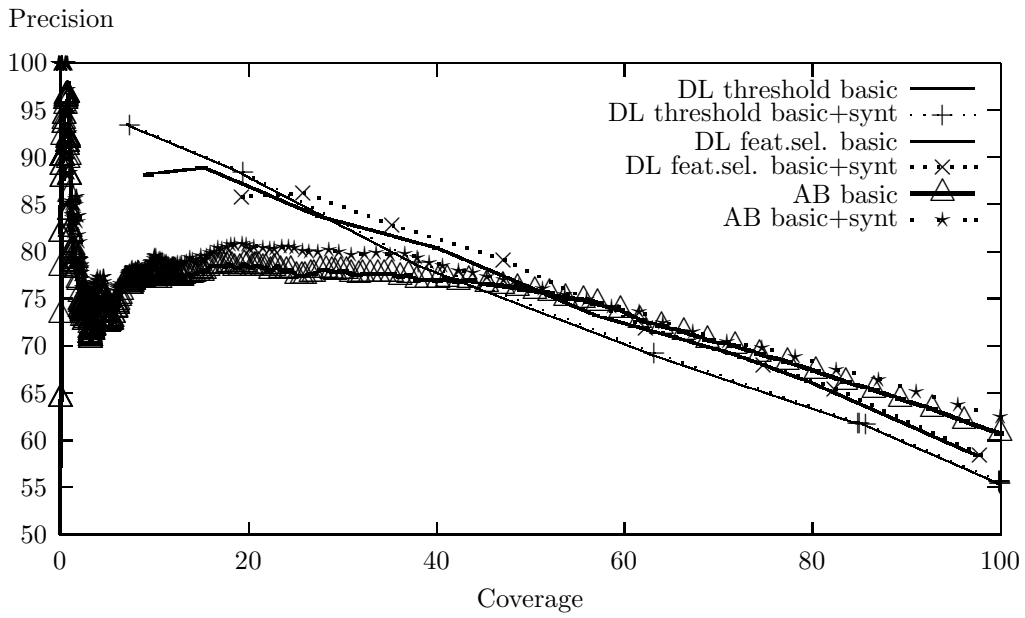

Figure IV.3: Graph of systems based on precision/coverage trade-off in Senseval-2 data.

The results of the three experiments are better illustrated in the precision/coverage graph in figure IV.7. The figure reveals an interesting behavior for different coverage ranges. In the high coverage range, AB on basic+syntactic features attains the best performance, which is consistant with the results in section IV.6. In the medium coverage area, the feature selection method for DL obtains the best results, also for basic+syntactic features. Finally, in the low coverage and high precision area the decision-threshold method for DL is able to reach precisions in the high nineties, with almost no profit from syntactic features.

The two methods to raise precision for DL are very effective. The decision-threshold method obtains constant increase in performance up to 93% precision with 7% coverage. The feature selection method attains 86% precision with 26% coverage using syntactic features, but there is no further improvement. In this case DL is able to obtain extremely good accuracy rates (at the cost of coverage) restricting to the use of the most predictive features. A possible improvement of this approach would be to combine the outputs of both DL methods, covering more cases with high precision. On the con-

trary, we have had problems in adjusting the AB algorithm for obtaining high-precision predictions.

The figure also shows, for coverage over 20%, that the syntactic features consistently allow for better results, confirming that syntactic features improve the results of the basic set.

IV.8 Semantic features

In this section we will explore the contribution of semantic features. The Semcor corpus provides the synsets for all the content words in the context. Using this information, we devised some experiments in order to know whether we can take advantage of this information. The idea is that if we could disambiguate some words in the context, these would provide additional clues to disambiguate other words in the same context.

The assumptions of this experiment are difficult to meet in a real setting. Therefore, we did not define the features as disambiguated collocations but as "bag-of-words", because we can suppose that it would be easier to disambiguate some indeterminate words in a near context than guessing exactly the synsets of the local context.

Thus, the semantic features we will use in the next experiments are based on the WordNet hierarchy. They will represent synsets, semantic files (see experiment on coarse senses in section III.5.6), and hypernyms of the words in the context, in a "bag-of-words" way. This is the complete list:

- Synset: Synsets of each word in the context.
- Semantic-file: Semantic files of each word in the context.
- Hypernym: Immediate hypernyms of the words in the context.
- Ancestor(3): Hypernyms of the words in the context, up to distance 3.
- Ancestor: Hypernyms of the words in the context, up to unique beginners.

As we will see in the next section, only the synsets and relatives of nouns have been used in some experiments.

PoS	Basic		+ Synsets		+ Sem. Files		+ Hypernyms		+ Anc. (3)		+ Anc.	
	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.
Adj.	82	100	82	100	84*	100	82	100	84	100	**85**	100
Adv.	**72**	100	**72**	100	**72**	100	70	100	69*	100	70	100
Nouns	80	99	80	99	80	100	**81**	100	80	100	80	100
Verbs	**58**	92	57	94	**58**	93	57	96	55*	97	56*	97
Overall	70	97	70	98	**71**	97	70	98	69*	99	69	99

Table IV.12: Results on the word-set A in Semcor adding semantic features to the basic set. Precision (Pr.) and coverage (Cov.) shown. The '*' character after the precision indicates that the difference with respect to the basic feature set is significant. The best precision per row is given in bold.

IV.9 Performance of semantic features

In order to use the Semcor corpus, we go back to the "Semcor&DSO" setting (cf. section III.3.3.1). For our first experiment, we applied the DL algorithm on the A word-set (cf. section III.3.1), adding the new features to the basic feature set described in section III.4.1. We added one feature type each time, and evaluated the precision and coverage. The results are illustrated in table IV.12. The best precision for each row (PoS and Overall) is given in bold. The '*' character after the precision figure indicates that the difference of the result with respect to the basic feature set is significant according to the Student's t-test (cf. section II.5.2).

The table shows that only semantic files improve the precision of the basic set, but according to the t-test, the 1-point difference is not significant overall. The test is positive only for adjectives. The other features do not improve the results. All the features based on hypernyms improve the coverage, but not the precision. Some reason for these low results are the following:

1. The DL algorithm does not take profit from these features because it focuses only in the best evidence. As we noticed in section IV.6, algorithms based on the combination of features, like AB, may be better suited to scale up from basic features.

2. The WordNet hierarchy is richer for nouns than from other categories. Therefore, one option is to use only the nouns in the context to define semantic features, in order to reduce the noise.

3. Local features obtain usually better precision than "bag-of-words" topical features (cf. section III.5.3). This implies that normally they have more weight in the decision, but there are some cases in which we may have to rely on the topical context (e.g.: the method in chapter VI, where examples are obtained automatically substituting the target word by a relative). Therefore, even if the results for the combination of basic features (local + topical) do not improve, it would be interesting to know whether the new features can improve the results of the topical set alone.

Taking these factors into account we designed new experiments. We did not explore exhaustively all the cases, because the low results of the tests made us think in a better approach for using the WordNet hierarchy for WSD: selectional preferences (section IV.10).

We report here the results obtained applying the NB algorithm (cf. section II.4), using only the nouns in the context, and adding the features to the topical set. For this setting we measured only the results with the hierarchical features (*hypernyms*, *ancestors(3)*, and *ancestors*). We also tested other combinations, but the differences in the results were small. We think that the results of this experiment are enough to reflect the contribution we can expect for this kind of semantic features.

Table IV.13 illustrates the precision and coverage achieved by NB for the different feature sets. The best precision for each line (PoS and overall) is given in bold. The '*' character indicates significance of the Student t-test. The results show that hypernym-based features improve the precision in one point, and according to the t-test, that difference is significant for the sets *Ancestor* and *Ancestor(3)*. We can see that the difference in overall precision is small; and only adjectives improve clearly their performance (2%-6% recall). We analyzed the results of the two adjectives in the set, and all the improvement was due to the word *long* (193 examples in Semcor). If we examine words with other PoS, we see that the differences are very low, and the t-test is negative in almost all cases.

All in all, the experiments suggest that other ways should be tried to benefit from these features. Instead of the "bag-of-words" approach, the use of dependency relations seems a better way to explore semantic generalization, as we will see in the next sections. However, the experiments were performed on Semcor, which means that there were few examples to train, but also that the system would be applicable to all the words that appear in Semcor. As

PoS	Topical		+ Hypernyms		+ Ancestor(3)		+ Ancestor	
	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.	Prec.	Cov.
Adj.	82	100	84	100	87*	100	**88***	100
Adv.	**67**	100	**67**	100	**67**	100	**67**	100
Nouns	79	100	**80***	100	**80**	100	**80**	99
Verbs	54	100	54	100	**55***	100	54	99
Overall	67	100	**68**	100	**68***	100	**68***	100

Table IV.13: Results with the NB method on the word-set A in Semcor, adding semantic features to the topical set. Only synsets of nouns used. The '*' character after the precision indicates that the difference with respect to the topical feature set is significant. The best precision per row is given in bold.

we have seen in the Senseval literature (cf. chapter II), the all-words systems perform significantly lower than lexical-sample systems, and it is not easy for them to overcome the MFS baseline.

IV.10 Learning of selectional preferences

Selectional preferences try to capture the fact that linguistic elements prefer arguments of a certain semantic class; e.g. a verb like *eat* prefers as object edible things, and as subject animate entities, as in, (1) *She was eating an apple*. Selectional preferences get more complex than it might seem. E.g. (2) *The acid ate the metal*, (3) *This car eats a lot of gas*, (4) *We ate our savings*, etc.

Corpus-based approaches for selectional preference learning extract a number of relations (e.g. verb/subject) from large corpora and use an algorithm to generalize from the set of nouns for each verb separately. Usually, nouns are generalized using classes (concepts) from a lexical knowledge base like WordNet.

IV.10.1 Selectional preference models

Before we describe our approach, we will explain the terminology we use. We say *concept* and *class* to refer to the synsets in WordNet. Synsets are represented as sets of synonyms, e.g.: {*food, nutrient*}. When a *concept* is taken as a *class*, it represents the set of synsets that are subsumed by this

synset in the hierarchy. A word sense in WordNet is a word-concept pairing, e.g.: given the concepts a={*chicken, poulet, volaille*} and b={*wimp, chicken, crybaby*} we can say that chicken has two word senses, the pair *chicken-a* and the pair *chicken-b*. In fact the former is sense 1 of chicken ({$chicken_1$}), and the later is sense 3 of chicken ({$chicken_3$}). For the sake of simplicity, we also say that {*chicken, poulet, volaille*} represents a word sense of chicken. We can see the four senses of *chicken* in WordNet 1.6 in figure IV.4.

Word: `chicken`

- Sense 1 => {`chicken, poulet, volaille`}
- Sense 2 => {`chicken, Gallus gallus`}
- Sense 3 => {`wimp, chicken, crybaby`}
- Sense 4 => {`chicken`}

Figure IV.4: Senses of *chicken* and corresponding synsets (concepts or classes) in WordNet 1.6 .

In our approach, the model is trained using subject-verb and object-verb associations extracted from Semcor. The syntactic relations were extracted using the Minipar parser. A peculiarity of this exercise is the use of a sense-disambiguated corpus, in contrast to using a large corpus of ambiguous words. This corpus makes it easier to compare the selectional preferences obtained by different methods. Nevertheless, the approach can be easily applied to larger, non-disambiguated corpora.

We have extended Resnik's selectional preference model (Resnik, 1992, 1997) from word-to-class (e.g. verbs - nominal concepts) to class-to-class (e.g. verbal concepts - nominal concepts). This model emerges as a result of the following observations:

- Distinguishing verb senses can be useful. The four examples for *eat* presented in the beginning of section IV.10 are taken from WordNet, and each corresponds to a different word sense: example (1) is from the *take in solid food* sense, (2) from the *cause to rust* sense, and examples (3) and (4) from the *use up* sense.
- If the word senses of a set of verbs are similar (e.g. word senses of ingestion verbs like *eat, devour, ingest,* etc.) they can have related

selectional preferences, and we can generalize and make a class of verbs share the same selectional preference.

Our formalization distinguishes among verb senses; that is, we treat each verb sense as a different unit that has a particular selectional preference. From the selectional preferences of single verb senses, we also infer selectional preferences for classes of verbs. For that, we use the relation between word senses and classes in WordNet.

Summarizing, we want to model the probability of a nominal concept given that it is the subject/object of a particular verb (*word-to-class*), a verb sense (*word sense-to-class*), or a verbal concept (*class-to-class*). We will now explain the three models in turn.

We have to notice that the models we will describe now do not represent probabilities in the strict sense, but heuristic weights that we will represent with the symbol W. This happens because in our model we will assume that a concept and its hypernym are independent, which is not real. This allows us to define easily weighted *class-to-class* relations for all concepts from a few tagged examples, but at the cost of losing the support of a well-established probabilistic distribution. The validity of our approach will be tested in the WSD exercise.

We will apply this notation in the description of the models: v stands for a verb, cn (cv) stands for a nominal (verbal) concept, cn_i (cv_i) stands for the concept linked to the i-th sense of the given noun (verb), *rel* could be any grammatical relation (in our case object or subject), $\subseteq$ stands for the subsumption relation, *fr* stands for frequency and $\hat{fr}$ for the estimation of the frequencies of classes (we will use estimated frequencies because of the sparse data in Semcor, the estimation method is presented in section IV.10.2).

The models will be illustrated with an example: the object relation between the nominal concept $\{chicken_1\}$ and the verb *eat*.

IV.10.1.1 Word-to-class model : $W(cn_i|rel\ v)$

The weight of the concept $\{chicken_1\}$ being the object of *eat* depends on the probabilities of the concepts subsumed-by and subsuming $\{chicken_1\}$ being objects of *eat*. For instance, if $chicken_1$ (first sense of *chicken*) never appears as an object of *eat*, but other word senses under its hypernym {*food, nutrient*} do, $W(\{chicken_1\}|object\ eat)$ will be higher than 0.

Formula IV.1 shows that for all concepts subsuming cn_i the probability of cn_i given the more general concept times the probability of the more general

concept being a subject/object of the verb is added. The first probability is obtained dividing the estimated class frequencies of cn_i with the estimated class frequencies of the more general concept. The second probability is calculated dividing the estimated frequency of the general concept occurring as object of *eat* with the number of occurrences of *eat* with an object. The estimation of the frequencies of the classes will be described in section IV.10.2.

$$W(cn_i|rel\, v) = \sum_{cn \supseteq cn_i} P(cn_i|cn) \times P(cn|rel\, v) = \sum_{cn \supseteq cn_i} \frac{\hat{f}r(cn_i, cn)}{\hat{f}r(cn)} \times \frac{\hat{f}r(cn\, rel\, v)}{fr(rel\, v)} \quad \text{(IV.1)}$$

IV.10.1.2 Sense-to-class model : $W(cn_i|rel\ v_j)$

Using a sense-tagged corpus, such as Semcor, we can compute the weight of the different senses of *eat* having as object the class $\{chicken_1\}$. We use the formula IV.1 for each sense of *eat* separately. In this case we have different selectional preferences for each sense of the verb (v_j): $W(cn_i|rel\ v_j)$.

IV.10.1.3 Class-to-class model : $W(cn_i|rel\ cv_j)$

We compute the weight of the verb classes associated to the senses of *eat* having as object $\{chicken_1\}$, using the probabilities of all concepts above $\{chicken_1\}$ being objects of all concepts above the possible senses of *eat*. For instance, if *devour* never appeared on the training corpus, the model could infer its selectional preference from that of its superclass {*ingest, take in*}.

Formula IV.2 shows how to calculate the W value. For each possible verb concept (cv) and noun concept (cn) subsuming the target concepts (cn_i, cv_j), the probability of the target concept given the subsuming concept (this is done twice, once for the verb, once for the noun) times the probability the nominal concept being subject/object of the verbal concept is added.

$$W(cn_i|rel\, cv_j) = \sum_{cn \supseteq cn_i} \sum_{cv \supseteq cv_j} P(cn_i|cn) \times P(cv_j|cv) \times P(cn|rel\, cv) = \sum_{cn \supseteq cn_i} \sum_{cv \supseteq cv_j} \frac{\hat{f}r(cn_i, cn)}{\hat{f}r(cn)} \times \frac{\hat{f}r(cv_j, cv)}{\hat{f}r(cv)} \times \frac{\hat{f}r(cn\, rel\, cv)}{fr(rel\, cv)} \quad \text{(IV.2)}$$

IV.10.2 Estimation of class frequencies

Frequencies for classes can be counted directly from the corpus when the class is linked to a word sense that actually appears in the corpus, written as $fr(cn_i)$. Otherwise they have to be estimated using the direct counts for all subsumed concepts, written as $\hat{fr}(cn_i)$.

Formula IV.3 shows the estimation for the nominal class cn. All the counts for the subsumed concepts (cn_i) are added, but divided by the number of classes for which cn_i is a subclass (that is, all ancestors in the hierarchy). This is necessary to guarantee the following:

$$\sum_{cn \supseteq cn_i} P(cn_i|cn) = 1$$

Formula IV.4 shows the estimated frequency of a concept given another concept. In the case of the first concept subsuming the second, it is equal to 0; otherwise the frequency is estimated as in formula IV.3.

Formula IV.5 estimates the counts for [nominal-concept relation verb] triples for all possible nominal-concepts, which is based on the counts for the triples that actually occur in the corpus. All the counts for subsumed concepts are added, divided by the number of classes in order to guarantee this relation:

$$\sum_{cn} P(cn|rel\ v) = 1$$

Finally, formula IV.6 extends formula IV.5 to [nominal-concept relation verbal-concept] in a similar way. We can see an example for $\hat{fr}(cn\, rel\, v)$ in figure IV.6.

IV.11 Evaluation of selectional preferences

The acquired preferences will be tested on a WSD exercise. Our goal in this experiment will be to choose the correct word sense for all nouns occurring as subjects and objects of verbs, but the method could also be used to disambiguate the verbs. The algorithm selects the word sense of the noun that is below the strongest nominal class for the verb, verb sense, or verb class (depending on the model). When more than one word sense is below the strongest class, all are selected with equal weight.

$$\hat{fr}(cn) = \sum_{cn_i \subseteq cn} \frac{1}{classes(cn_i)} \times fr(cn_i) \tag{IV.3}$$

$$\hat{fr}(cn_i, cn) = \begin{cases} \sum_{cnj \subseteq cni} \frac{1}{classes(cn_j)} \times fr(cn_j) & if\ cn_i \subseteq cn \\ 0 & otherwise \end{cases} \tag{IV.4}$$

$$\hat{fr}(cn\, rel\, v) = \sum_{cn_i \subseteq cn} \frac{1}{classes(cn_i)} \times fr(cn_i\, rel\, v) \tag{IV.5}$$

$$\hat{fr}(cn\, rel\, cv) = \sum_{cn_i \subseteq cn} \sum_{cv_i \subseteq cn} \frac{1}{classes(cn_i)} \times \frac{1}{classes(cv_i)} \times fr(cn_i\, rel\, cv_i) \tag{IV.6}$$

Figure IV.5: Estimation of frequencies.

- Occurrences of *eat* in the corpus:
  - *eat* - object - $\{chicken_1, ...\}$
  - *eat* - object - $\{pork_1, ...\}$
- Subsumption relations from WordNet:
  - $\{chicken_1, ...\} \subseteq \{food, nutrient, ...\}$
  - $\{pork_1, ...\} \subseteq \{food, nutrient, ...\}$
- Estimation table:

	fr	$\hat{fr}$
$\{chicken_1, ...\}$	1	0.5
$\{pork_1, ...\}$	1	0.5
$\{food, nutrient, ...\}$	0	1

Figure IV.6: Example of estimation of frequency for the case $\hat{f}$r(cn rel v); where rel = *object*, v = *eat*, and $cn = \{food, nutrient, ...\}$.

In order to apply the method, we need a corpus to learn the preferences, and another for testing. We can use an untagged corpus for learning, introducing all the possible tags for each relation extracted. For this approach to be useful, we would require a big corpus, in order to reduce the noise of the ambiguity. This is the usual approach for selectional preference learning in the literature. Another option is to use a corpus of sense-tagged relations, but such a corpus is difficult to obtain. We already have the Semcor corpus available, and we have used it in previous experiments, therefore we opted for this approach. We used Semcor to learn the preferences, and also for testing (via cross-validation).

The fact that we rely on a general all-words corpus, and not in a list of tagged instances for each target word makes it difficult to compete with supervised systems, and even with the MFS baseline. This method is closer to the unsupervised approach, and should be seen in that light.

Two experiments will be described in this section. We will rely on the "Semcor&DSO" setting (cf. section III.3.3.1). For the lexical-sample, we will use only the 8 nouns in the set A (cf. section III.3.1.1). We will apply 10 fold cross-validation, learning the preferences from 90% of the data, and disambiguating the remaining 10% for each iteration. For the all-nouns experiment, four files previously used in the baseline experiments (cf. section III.5.7) were disambiguated. In this case, to disambiguate each file, we trained the selectional preferences on the rest of Semcor.

As we said, only nouns occurring as subjects and objects of verbs can be disambiguated. We can see in table IV.14 the proportion of examples where the target nouns have been marked as subject or object (for the lexical sample experiment). Note that only 19% of the occurrences of the nouns are objects of any verb, and 15% are subjects. This implies that the method by itself is not enough for full-coverage WSD. In order to extend the model we may:

- Use other relations besides subject and object.
- Integrate this knowledge with other information sources.
- Use an alternative parser that could help to detect more relations. We have observed that many object/subject are not identified.

Nevertheless, the following experiments will show us whether the information learned in the form of selectional preferences can be another valid source of knowledge to be integrated in a WSD system. The other test we

Nouns	Senses	Freq.	Object		Subject	
			Freq.	%	Freq.	%
account	10	27	8	29.6	3	11.1
age	5	104	10	9.6	9	8.7
church	3	128	19	14.8	10	7.8
duty	3	25	8	32.0	1	4.0
head	30	179	58	32.4	16	8.9
interest	7	140	31	22.1	13	9.3
member	5	74	13	17.6	11	14.9
people	4	282	41	14.5	83	29.4
Overall	67	959	188	19.6	146	15.2

Table IV.14: Frequencies and polisemy of the lexical sample nouns, together with the relations extracted from Semcor by Minipar.

Method	Object			Subject		
	Prec.	Cov.	Rec.	Prec.	Cov.	Rec.
Random	19.2	100	19.2	19.2	100	19.2
MFS	69.0	100	69.0	69.0	100	69.0
Word2class	66.9	86.7	58.0	69.8	79.4	55.4
Class2class	65.7	97.3	64.0	68.3	98.6	67.3

Table IV.15: Performance of selectional preference models on the nouns from set A in Semcor. Random and MFS baselines, and two models: word-to-class and class-to-class.

will make is to measure whether the extended class-to-class model is able to generalize well and improve the results of the word-to-class model.

Table IV.15 shows the overall results for the lexical sample experiment. Together with the two baselines (random and MFS), the precision, coverage, and recall of the word-to-class model and the class-to-class are given[1].

The classic word-to-class model gets slightly better precision than class-to-class, but class-to-class is near complete coverage and thus gets the best recall. This indicates that the algorithm is able to generalize well and learn useful information. The recall is above the random baseline, but slightly below MFS. We have to notice that with so few points of data, the MFS baseline is difficult to beat. Another factor is that there is no smoothing or

[1]We decided not to include the sense-to-class model, because it requires a sense-tagged corpus, and at this point we think that it is more interesting to study the performance of the class-to-class model, which is learnable from untagged corpora and can obtain preferences for verb senses not seen in the corpus.

Word	Object			Subject		
	Freq.	Prec.	Cov.	Freq.	Prec.	Cov.
account	8	37.5	100	3	33.3	100
age	10	77.8	90.0	9	66.7	100
church	19	63.2	100	10	60.0	100
duty	8	25.0	100	1	0	100
head	58	74.1	100	16	56.2	100
interest	31	55.2	93.5	13	15.4	100
member	13	38.5	100	11	36.4	100
people	41	82.1	95.1	83	86.4	97.6
Overall	188	65.7	97.3	146	68.3	98.6

Table IV.16: Results of the class-to-class model per word in Semcor (set A).

Method	Object			Subject		
	Prec.	Cov.	Rec.	Prec.	Cov.	Rec.
Random	26.5	100	26.5	29.6	100	29.6
MFS	69.8	100	69.8	79.0	100	79.0
W2C	51.7	80.1	41.4	69.9	85.6	59.8
C2C	53.2	95.0	50.5	70.5	98.1	69.2

Table IV.17: Performance of selectional preference models on the nouns from 4 Semcor files.

cut-off value involved, which forces the algorithm to decide with low confidence. Table IV.16 shows the results for the class-to-class model per word, where we can see the number of examples for each target word, the precision, and the coverage. Note that the system is tested using cross-validation, and with these amounts of examples, most of the time the preferences have to be generalized using words different to the target. We can see that normally better precision is obtained for words with higher number of examples, like *people* or *head*.

To conclude this section, table IV.17 illustrates the results of the selectional preference models in 4 Semcor files. The averaged performance values are given for the two baselines and the two models. We see that the class-to-class model obtains better precision and recall than the word-to-class model, showing that it is better suited for this task. However, the loss with respect to the MFS baseline is bigger than in the previous experiment.

IV.12 Conclusions

We have performed several experiments throughout the chapter for different types of features. We will first describe the conclusions derived from each group of experiments, and finally we will summarize our overall conclusions.

Syntactic features on Semcor and DSO

Our first experiments were performed on Semcor and DSO, with the DL method. In Semcor, the syntactic features achieve lower performance (in precision and coverage) than the basic set, and the combination of the two sets (basic and syntactic) does not help to improve the results. In DSO there is a small improvement adding the syntactic features to the basic set (0.3% precision, 0.1 %coverage). Verbs are the most benefited from syntactic knowledge (0.5% precision, 0.1% coverage). Taken separately, the syntactic feature set improves the precision of the basic features, but the coverage is still low.

In order to know the reason for this low performance (specially in Semcor), we analyzed the behavior of the different features separately. We applied the DL algorithm using only one feature each time, and we evaluated the precision and coverage of each piece of evidence. We observed that the syntactic features achieved good precision, but the recall was very low in comparison with basic features, and they could be applied only in a few cases. However, some syntactic features achieved comparatively good recall for verbs, specially ngrams, suggesting that some subcategorization information had been acquired.

For further analysis, we focused on some words in the Semcor experiment, and analyzed the learned decision lists. These are the main conclusions:

1. Syntactic features usually have fewer occurrences in the training corpus than basic features, and they rank low in the decision lists. Even syntactic features with high frequency in training usually have basic features above them, which suggest that the information may be redundant.

2. In the case of words with a dominant sense, some syntactic features that appear frequently and do not carry much information (e.g. the presence of a determiner linked to a noun) can introduce noise and

point strongly to the most frequent sense. This happens also with non-syntactic features, but in a less harming scale because they comprise a more reduced and controlled set.

3. The parser fails to detect many dependencies and commits some errors, which affects the precision and specially the coverage.

Syntactic features on the Senseval-2 corpus

For our next setting, the Senseval-2 dataset, we decided to apply also another ML method: AB. We expected that this would alleviate the effect of the two former problems in the previous experiments on syntactic features. But first, we analyzed the performance of the syntactic features with DLs. Surprisingly, the results were significantly better in this corpus. Syntactic features alone obtained better F1 value than the MFS baseline. The F1 value was much higher in this experiment than in the DSO task, even when the recall of the MFS baseline was higher in DSO.

In our next experiment, we tested the combination of basic and syntactic features using the two ML methods. We extracted these conclusions:

- AB is able to outperform DL in all cases, except for local features.
- Syntactic features get worse results than local features.
- Syntactic features prove to be useful in the combination. DLs profit from the additional syntactic features but the difference is only statistically significant for verbs. On the other hand, AB attains significant improvement (1.8% overall, 2.7% for verbs).

Syntactic features and high precision systems

Finally, we tested the effect of syntactic features for high precision WSD. We analyzed two systems based on DL (feature selection and decision-threshold), and one based on AB (decision-threshold). These are the main observations:

- Syntactic features help to improve the F1 result of the basic set in all cases.

- Adjusting the methods to a minimum loss of coverage (discarding the most difficult testing examples), the overall F1 improves for the three methods.

- The methods based on DL reach 93% precision at 7% coverage (decision-threshold), and 86% precision at 26% coverage (feature selection). Syntactic features are specially helpful for feature selection.

- AB does not achieve high precision figures, but it obtains the highest F1 score in this setting, with 66.7% precision and 84.5% coverage.

Semantic features

Experiments on these features, which were based on synsets of the words in the context, did not achieve good performance. This feature set was defined using the WordNet hierarchy, and the information from the semantic-files. The experiments were performed on Semcor, which means that there were few examples to train, but also that the system would be applicable to all the words that appear in Semcor. As we have seen in the Senseval literature (cf. chapter II), the all-words systems perform significantly lower than lexical-sample systems, and it is not easy for them to overcome the MFS baseline.

The results show that overall, the system is able to improve the performance of the topical feature set, using the NB algorithm. This could be useful when the local contexts are not reliable, as could happen with automatically acquired features (cf. chapter VI). Another case where the recall is improved is for adjectives, with a gain of 3% recall.

All in all, the experiments suggest that other ways should be tried to benefit from these features. Instead of the "bag-of-words" approach, the use of dependency relations seems a better way to explore semantic generalization.

Selectional preferences

We tested whether selectional preference learning could give us a better way to use semantic information for WSD. The experiments had the following characteristics:

- Extract object/subject relations between nouns and verbs, applying the Minipar parser.

- Learn preferences from the WordNet hierarchy using two models: word-to-class, and class-to-class.
- Disambiguate nouns in the Semcor corpus, via cross-validation.

Our first observation was that the coverage of the subjects and objects was very low. Some ways to address this would be by extracting other types of relations, integrating selectional preferences in a system with other types of features, or using a more accurate parser. However, we performed the disambiguation for the examples that we could obtain, to measure whether this information was useful. The two experiments, which were performed for a sample of nouns, and for all the nouns in four Semcor files, took us to the following conclusions:

- The class-to-class model obtains better recall than the word-to-class model, with only a small loss in precision. Class-to-class learns selectional preferences for senses of verbs that do not occur in the corpus, via inheritance.
- The recall of the class-to-class model gets close to the MFS baseline. We have to note that this is a hard baseline for this kind of all-words systems, as we have seen in our study of the literature (cf. section IV.2.2).
- The preferences are acquired from a small set of tagged examples, and for some words the results are very low. The words with more examples to train seem to have better performance.

Apart from the low coverage, another limitation of this approach is that no cut-off values or smoothing is applied, and the algorithm is forced to make decisions with few data. Applying a threshold could help to improve precision. Another way we would like to explore is the use of a big untagged corpus to learn the preferences. We are also interested in the performance when disambiguating words with other PoS than nouns. Finally, we would like to test these selectional preferences in combination with other feature types, like the ones we have been exploring previously. We think that despite their low coverage, selectional preferences would help to improve the overall performance of the system, although it is not straightforward how to integrate them in a supervised system. One possibility would be to include the sense chosen by the selectional preference model in the feature set, in a fashion similar to (Stevenson and Wilks, 1999).

Overall conclusions

The goal of this chapter has been to analyze richer features, in order to know whether the effort of extracting this knowledge is useful for WSD. The types of features we have analyzed in this chapter are divided in three groups: syntactic features, semantic features, and selectional preferences.

For syntactic features, the results show that basic and syntactic features contain complementary information, and that they are useful for WSD. The contribution of this type of feature is specially noticeable for the algorithm AB in the standard setting, and for DLs when applying the precision/coverage trade-off.

Regarding semantic features, we have seen that they could contribute slightly to improve the performance of an all-words system. However, the "bag-of-words" approach does not seem to benefit much from the WordNet hierarchy. Instead, the generalization of syntactic dependencies using WordNet offers promising results, as has also been seen in[2](Mihalcea and Faruque, 2004). Improved performance could come from integrating selectional preferences together with other feature types, but this path has not been explored in this dissertation.

[2]The system "SenseLearner" has been described in section II.8.

V. CHAPTER

Sparse data problem and smoothing techniques

V.1 Introduction

In the previous chapter we have studied the contribution of different knowledge sources to WSD. Now we will focus on the sparse data problem, which affects several NLP techniques that estimate probabilities from real texts, like statistical MT or text categorization. Both for NLP and WSD, most of the events occur rarely, even when large quantities of training data are available. In supervised WSD, the difficulty of building a hand-tagged corpus makes the sparse data problem one of the main barriers to achieve higher performance figures. Normally, for each word there is only a handful of occurrences with sense tags. For example, if we take the word *channel*, we see that it appears 5 times in SemCor, one of the few sense-tagged corpus for all-words: the first sense has four occurrences, the second a single one, and the other 5 senses are not represented. For a few words, more extensive training data exists. Senseval-2 (Edmonds and Cotton, 2001) provides 145 occurrences of *channel*, but still some of the senses are represented by only 3 or 5 occurrences.

Moreover, the fine-grained analysis of the context performed by most WSD systems requires that we represent it by means of many features, some of them rare. The occurrences of these features can be very informative, and the estimation of rare-occurring features might be crucial to have high

performances. These scenarios, where the dimensionality of the feature space exceeds the number of examples, show a big potential for overfitting.

Smoothing refers to the techniques that try to estimate the probability distribution that approximates the one we expect to find in held-out data. In WSD, if all occurrences of a feature for a given word occur in the same sense, the Maximum Likelihood Estimation (MLE) would give a 0 probability to the other senses of the word given the feature, which is a severe underestimation. We will denote these cases as X/0, where X is the frequency of the majority sense, and zero is the frequency of the other senses.

For instance, if the word *Jerry* occurs in the context of *art* only once in the training data with a given sense, does it mean that the probability of other senses of *art* occurring in the context of *Jerry* is 0? We will see in section V.6.3 that this is not the case, and that the other senses are nearly as probable. Our smoothing study will show for this feature of the word *art* that the smoothed ratio should be closer to 1/1.

In this chapter, we follow the smoothing method proposed by Yarowsky in his PhD dissertation (Yarowsky, 1995a), and present a detailed algorithm of its implementation for the WSD problem, defining some of the parameters used, alongside the account of its use by three different ML algorithms: DL, NB, and VSM (cf. section II.4). The impact of several smoothing strategies is also presented, and the results indicate that the smoothing method explored in this work is able to make both statistically motivated methods (DL and NB) perform at very high precisions, comparable and in some cases superior to the best results attained in the Senseval-2 competition (cf. section II.7). We also show that a simple combination of the methods and a fourth system based on SVM (cf. section II.4) attains the best result for the Senseval-2 competition reported so far. This system was submitted to the Senseval-3 competition, obtaining one of the top scores (cf. section II.8).

Another motivation for this work is the possibility to use smoothing techniques in bootstrapping approaches. Bootstrapping techniques such as (Yarowsky, 1995b) have shown that having good seeds, it would be possible to devise a method that could perform with quality similar to that of supervised systems. Smoothing techniques could help to detect rare but strong features, which could be used as seeds for each of the target word senses. In the next chapter, we will apply the method presented in (Leacock *et al.*, 1998) to obtain examples automatically by means of the WordNet hierarchy. This method could be extended relying on smoothing to obtain relevant features for each sense, and using these features as the source of new examples.

This chapter is organized as follows. Related work on smoothing and ensembles of algorithms is summarized in section VII.2. Section V.3 presents the experimental setting, and the feature set is described in section V.4. Section V.5 introduces smoothing of features, and section V.6 presents the specific algorithm with examples. Section V.7 presents the evaluation and comparison with other systems in Senseval-2, and section V.8 gives the results of the official Senseval-3 evaluation. Finally, the last section draws some conclusions.

V.2 Related work

Hoste *et al.* (2002) observe that parameter optimization is a key factor in WSD performance. They note that there are many interactions among the feature space, the learning examples, and the parameters of the ML method. The optimization of these interactions per word would lead to better results for a given algorithm. Our results show that it is indeed the case, and weaker learning algorithms such as DL, NB and VSM attain performances close or superior to SVM with the help of appropriate smoothing techniques.

An important parameter that has to be estimated previously for some disambiguation methods is the smoothing of feature frequencies. Algorithms like DL or NB cannot handle 0 probabilities for a sense given a feature. Although these algorithms have been widely used in the literature (specially in combination with other methods, as we will describe below), there are few works that address this problem with specific techniques, and normally simple default values are used.

Yarowsky (1995a), in his dissertation work, provided a study on ways to estimate the distribution of each different collocation in the model. His method is based on the mean value of accuracy in held-out data. In order to better estimate X/0 and X/1 frequencies, the occurrences in held-out data are counted; the basic idea is to assume that all collocations with the same sense distribution in the primary training data have the same true distribution. This approach is further refined grouping features for the same feature type, target PoS, etc. and using log-linear interpolation with the observed points. He showed that if primary training, test data, and held-out data are similar, then the mean distribution will be a better estimation than raw frequencies. This approach applies ideas from other smoothing methods in the literature: the Good Turing algorithm (Good, 1953), and the Method

of Deleted Estimation (Jelinek and Mercer, 1980).

Good (1953) merges all the distributions that have the same raw frequency in held-out data and estimates the smoothed probability applying a measure directly on those counts. This method is normally used in combination with others. The Method of Deleted Estimation (Jelinek and Mercer, 1980), linearly interpolates higher-order values with lower-order models. The probability of a feature would depend on the probabilities of the components in some extent, which is defined by the parameter λ. This method also uses held-out data to estimate probabilities. From a broader perspective, Chen (1996) provides a comprehensive description of diverse smoothing methods, and its application to different NLP problems. He implements the above described Method of Deleted Estimation, Katz smoothing (Katz, 1987), and Church-Gale smoothing (Church and Gale, 1991); and compares them to two new methods developed for his dissertation.

Regarding the relation between smoothing and ML methods for WSD, in (Ng, 1997), NB was applied using a simple method of smoothing, where zero counts were replaced by the probability of the given sense divided by the total number of examples. This approach has been followed in other experiments with NB (Escudero *et al.*, 2000b). For our work, we used this method as baseline, and also in combination with the method we will describe in section V.6. In a more recent work (Lee and Ng, 2002), 4 ML methods (NB, AB, SVM, and DTrees) are applied separately to the Senseval-2 English Lexical Sample data. For NB, the probabilities are smoothed using a simple method (Laplace, "add one"). They report better results with SVM than the best Senseval-2 result (65.4% vs. 64.2%). They have not attempted to combine the different methods, and no parameter estimation has been performed for the individual classifiers.

Together with the smoothing algorithm, in this chapter we will also test the integration of different ML methods in combined systems, which has been shown to be one of the most successful approaches in the Senseval competitions. We already described in section II.7 the JHU-English system (Yarowsky *et al.*, 2001), which consisted on voting-based classifier combination, and obtained the best performance in the English lexical-sample task. In their training models, they assign weights to different features, depending on the type of feature and the distance to the target word. This system is further refined in (Cucerzan and Yarowsky, 2003), including new algorithms like the Mixture Model (MM), and applying a filtering process to identify the relevance of surrounding words to disambiguate the target. This last

paper reports a significative increase of the results for NB and Mixture Models when using feature-weights and filtering. The final results outperform in 2.3% the best Senseval-2 submission (66.5% Vs 64.2%). In section V.7, some of these latter methods have been compared with our final algorithm for evaluation in the Senseval-2 setting.

Finally, another approach is the combination of different linguistic knowledge sources to disambiguate all the words in the context, as in (Stevenson and Wilks, 2001). In this work, they integrate the answers of three "partial taggers" based on different knowledge sources in a feature-vector representation for each sense. The vector is completed with information about the sense (including rank in the lexicon), and simple collocations extracted from the context. The TiMBL memory-based learning algorithm is then applied to classify the new examples. The "partial taggers" apply the following knowledge: 1) Dictionary definition overlap (optimized for all-words by means of simulated annealing), 2) Selectional preferences (based on syntactic dependencies and LDOCE codes), and 3) Subject codes from LDOCE applying the algorithm by (Yarowsky, 1992).

V.3 Experimental setting

For the main experiments in this chapter we applied the "Senseval2" setting (cf. section III.3.3.3), with a preprocessing stage for the multiwords and phrasal verbs (process described in section III.3.2). The preprocess is a necessary step in order to achieve competitive performance with other systems on the Senseval-2 lexical-sample task.

We relied on different ML methods in order to test the effect of the smoothing techniques: DL, NB, VSM, and SVM. We also constructed an ensemble of systems (by voting) to see how good was the final system in the Senseval framework.

We used the training part of the Senseval-2 corpus with cross-validation to estimate the C parameter for the SVM algorithm, and to obtain the smoothed frequencies for the features. For the set of experiments in evaluation, the systems are trained on the training part, and tested on the testing part.

Finally, we report here our results in the Senseval-3 competition using the approach presented in this chapter. The "Senseval3" experimental setting is given in section III.3.3.5.

V.4 Features

From the experience of the two previous chapters, we defined a new feature set that included syntactic dependency information. We also introduced features not tested previously, as the previous noun/verb/adj/adv in the sentence. With this augmented feature set we expected to obtain more profit of the smoothing procedure. The features can be grouped in four main sets:

Local collocations: Bigrams and trigrams formed with the words around the target. These features are constituted with lemmas, word-forms, or PoS tags[1]. Other local features are those formed with the previous/posterior lemma/word-form in the context for each main PoS. E.g. The feature *"prev_V_lem stand"* would indicate that the target word is preceded by the verb *stand*.

Syntactic dependencies: Syntactic dependencies were extracted using heuristic patterns, and regular expressions defined with the PoS tags around the target[2]. The following relations were used: object, subject, noun-modifier, preposition, and sibling. E.g. *list OBJ petition*.

Bag-of-words features: We extract the lemmas of the content words in the whole context, and in a ±4-word window around the target. We also obtain salient bigrams in the context, with the methods and the software described in (Pedersen, 2001). e.g. the feature *context_bigr visionary eyes* would express that *visionary eyes* has been found to be relevant for the target word, and has been seen in the given context.

Domain features: The WordNet Domains resource was used to identify the most relevant domains in the context. Following the relevance formula presented in (Magnini and Cavagliá, 2000), we defined 2 feature types: (1) the most relevant domain, and (2) a list of domains above a predefined threshold[3]. Other experiments using domains from SUMO, the EuroWordNet top-ontology, and WordNet's Semantic Fields were performed, but these features were discarded from the final set. The domain features were only used for the Senseval-3 experiments.

[1]The PoS tagging was performed with the fnTBL toolkit (Ngai and Florian, 2001).

[2]This software was kindly provided by David Yarowsky's group, from Johns Hopkins University.

[3]The software to obtain the relevant domains was kindly provided by Gerard Escudero's group, from Universitat Politecnica de Catalunya

V.5 Feature-type smoothing

We have already seen in the introduction that estimating X/0 features with MLE would yield a probability $P(s|f) = 1$ for the majority sense and a probability $P(s|f) = 0$ for the minority senses, which is an underestimation. Features with X/0 counts are usual when the training data is sparse, and these values must be smoothed before they are fed to some learning algorithms, such as DL or NB, as they lead to undetermined values in their formulations.

Other distributions, such as X/1, X/2, ... can also be estimated using smoothing techniques. Yarowsky (1995a) argues that the probability of the second majority sense in X/1 distributions would be overestimated by MLE.

For intermediate cases, such as X/2, X/3, etc. it is not clear whether the effort of modeling would be worth pursuing. For higher frequencies, using the raw frequency could be good enough. In this work we focused in X/0 and X/1 distributions.

The smoothing algorithm shown here (which we will call *feature-type smoothing*) follows the ideas of Yarowsky (1995a). The main criteria to partition the training data has been to use raw frequencies and feature types (e.g. *prev_N_wf*, feature type that represents the first noun word-form to the left of the target). Raw frequency is the most important parameter when estimating the distribution, and joining features of the same type is a conservative approach to partition the data. Therefore we join all occurrences of the *prev_N_wf* feature type that have the same frequency distribution for the target word, e.g. 1/0. This way, we perform smoothing separately for each word.

We could use the smoothed values calculated in this manner directly, but many data points would still be missing. For instance, when studying *prev_N_wf* in the X/0 frequency case for *art*, we found occurrences of this feature type in held-out data in the 1/0, 2/0 and 3/0 cases, but not the rest (4/0 and higher). In this case it is necessary to use interpolation for the missing data points, and we applied log-linear interpolation. The interpolation also offers additional benefits. Firstly, using the slope of the interpolated line we can detect anomalous data (such as cases where 1/0 gets higher smoothed values than 5/0) as we always expect a positive slope, that is, higher ratios deserve higher smoothed values. Secondly, interpolation can be used to override a minority of data points which contradict the general trend. These

points will be illustrated in the examples presented in section V.6.3.

However, when using interpolation, we need at least two or three data points for all feature types. For feature types with few points, we apply a back-off strategy: we join the available data for all words in the same Part of Speech. The rationale for this grouping is that strong features for a noun should be also strong for other nouns. In order to decide whether we have enough data for a feature type or not, we use the number of data points (minimum of three) available for interpolation. In order to check the validity of the interpolation, those cases where we get negative slope are discarded.

V.6 Feature-type smoothing algorithm

There are two steps in the application of the smoothing algorithm to the disambiguation task. First, we use the available training data in cross-validation, with an interpolation method, in order to estimate the smoothing tables for each feature type with X/0 or X/1 raw frequency. Second, the interpolated tables are accessed on the disambiguation phase, when the WSD methods require them. Sections V.6.1 and V.6.2 present the algorithms, and section V.6.3 shows some illustrative examples.

V.6.1 Building smoothing tables

We build two kinds of smoothing tables. The first kind is the application of the grouping strategy based on feature types and frequency distributions. Two tables are produced: one at the word level, and another at the PoS level, which we will call *smoothed tables*. The second kind is the result of the interpolation method over the two aforementioned tables, which we will call *interpolated tables*. All in all, four tables are produced in two steps for each frequency distribution (X/0 and X/1).

1) Construct smoothing tables for each target word and for each PoS. For each feature type (e.g. *prev_N_wf*), we identify the instances that have X/0 or X/1 distributions (e.g. *prev_N_wf Aboriginal*) and we count collectively their occurrences per sense. We obtain tables with (X',Y') values for each word, feature type and pair (X,Y); where (X,Y) indicate the values seen for each feature in the training part, and (X',Y') represent the counts for all the instances of the feature type with the same (X,Y) distribution in the held-out part.

```
1.  Construct word smoothing tables for X/0 (X0)
- For each fold from training-data (5 folds)
  Build count(f, w, sense) for all senses from the estimation-folds (4 folds)
  For each word w, for each feature f in each occurrence in target-fold (1 fold)
     get count'(f, w, sense) for all senses of w in target-fold
     If distribution of count'(f, w, sense) is of kind X/0 (X0) then
        For each sense
           if sense = s.max_s count(f, w, s)
           then            # sense is major sense in estimation-fold
              increment X' in table_word_X0(w,type(f),X)
           else
              increment Y' in table_word_X0(w,type(f),X)

- Normalize all tables:  X' is set to X, and Y' := Y'X'/X
  Output (No need to keep X'): normtable_word_X0(w, type(f), X) := Y'

2.  Log linear Interpolation
- Accumulate X' and Y' values
- Map into linear space:
  logtable_word_X0(w, type(f), X) :=
     log(acctable_word_X0(w, type(f), X).X'/acctable_word_X0(w, type(f), X).Y')
- Do linear interpolation of logtable:  sourcepoint(w, type(f)) = a0,
     gradient(w, type(f)) = a1
- For each X from 1 to ∞
     interpolatedtable_word_X0(w, type(f), X) := X/(e^(a0+a1 X))
```

Figure V.1: Construction of smoothing tables for X/0 features for words. The X/1 and PoS tables are built similarly.

We perform this step using 5-fold cross-validation on the training data. We separate in a stratified way[4] the training data in two parts: estimation-fold (4/5 of the data) and target-fold (1/5 of the data), which plays the role of the held-out data. We run the algorithm five times in turn, until each part has been used as target. The algorithm is described in detail in Figure V.1 for the X/0 case (the X/1 case is similar). Note that the X count corresponds to the majority sense for the feature, and the Y count to all the rest of minority senses for the feature. For example, we can see in the held-out columns in table V.1 the (X',Y') counts obtained for the feature type *prev_N_wf* and the target word *art* in the Senseval-2 training data for the X/0 cases.

2) Create interpolation curves. From the smoothing tables, we interpolate curves for feature types that have at least 3 points. The process is described in detail in the second part of Figure V.1. We first accumulate the

[4]By stratified, we mean that we try to keep the same proportion of word senses in each of the 5 folds.

Original		Held-out			Accumulated				Interpolated			
X	Y	X'	Y'	X'/Y'	X'	Y'	X'/Y'	log(X'/Y')	X"	Y"	X"/Y"	log(X"/Y")
1	0	4	4	1	4	4	1.00	0.00	1	0.91	1.10	0.09
2	0	6	1	6	10	5	2.00	0.69	2	1.18	1.69	0.52
3	0	2	0	∞	12	5	2.4	0.88	3	1.14	2.63	0.96
									4	0.98	4.08	1.40
									...			

Table V.1: Smoothing table for the feature *prev_N_wf* and the word *art* (X/0 distribution).

counts in the smoothed table from the previous step. The "Accumulated" columns in table V.1 show these values, as well as the X/Y ratio and its logarithm. The Y value is then normalized, and mapped into the logarithmic space. We apply a common linear interpolation algorithm called *least square method* (Neter *et al.*, 1985), which yields the starting point and slopes for each interpolation table. If we get a negative slope, we discard this interpolation result. Otherwise, we can apply it to any X, and after mapping again into the original space we get the interpolated values of Y, which we denote Y". Table V.1 shows the Y" values, the X"/Y" ratios, and the log values we finally obtain for the *prev_N_wf* example for *art* for $X = 1..4$ and $Y = 0$ ("Interpolated" columns). The X"/Y"ratios indicate that for X values lower than 4, the feature type is not reliable, but for $X >= 4$ and $Y = 0$, this feature type can be used with high confidence for *art*.

V.6.2 Using the smoothed values

The process to use the smoothed values in testing is described in Figure V.2. There we see that when we find X/0 or X/1 distributions, the algorithm resorts to the *obtain_smoothed_value* function to access the smoothing tables. The four tables constructed in the previous section are all partial, i.e. in some cases there is no data available for some of the senses. The tables are consulted in a fixed order: we first check the interpolated table for the target word; if it is not available for the feature type, we access the interpolated table for the PoS of the target word. Otherwise, we resort to the non-interpolated smoothing table at the word level. Finally we access the non-interpolated smoothing table for the PoS.

In cases were the four tables fail to provide information, we can benefit from additional smoothing techniques. The three ML methods that we have

```
Given an occurrence of a word w in testing, for each feature f in the context:
   Get count(f, w, sense) for all senses from all training (all 5 folds)
   If counts are not X/1 or X/0 then
      For each sense:
         count'(f, w, sense) := count(f, w, sense)

   Elseif count is X/Y (where Y is 1 or 0) then
      If Y' = obtain_smoothed_value(X, Y)
      Then
         For each sense
            If sense = s.max_s count(f, w, s) then        #(MAJOR SENSE)
               count'(f, w, sense) = X
            Elsif sense = 2nd_sense then #(ONLY IF Y=1, WHERE A MINORITY SENSE
                                                          OCCURS ONCE)
               count'(f, w, sense) := Y' #(SECOND SENSE GETS MORE CREDIT)
            Else
               count'(f, w, sense) := Y'/|othersenses| # (DISTRIBUTE WEIGHT UNIFORMLY
                                                  AMONG MINOR SENSES)

      Else               # (THERE IS NO SMOOTHING DATA FOR THIS X/Y)
         DISCARD                          #(THIS IS POSSIBLE FOR DL)
         For each sense
            If sense = s.max_s count(f, w, s) then       # (MAJOR SENSE)
               count'(f, w, sense) := X
            Elsif sense = 2nd_sense then #(ONLY IF Y=1, WHERE A MINORITY SENSE
                                                          OCCURS ONCE)
               count'(f, w, sense) := 1 # (SECOND SENSE GETS MORE CREDIT)
```

Figure V.2: Application of Feature-type smoothing to DL, NB and VSM.

applied have different smoothing requirements, and one of them (NB) does need a generally applicable smoothing technique:

DL: as it only uses the strongest piece of evidence, it can discard X/0 features. It does not require X/1 smoothing either.
NB: It needs to estimate all single probabilities, i.e. all features for all senses, therefore it needs smoothing in X/0, X/1 and even X/2 and larger values of Y. The reason is that in the case of polisemy degrees larger than 2, the rare senses might not occur for the target feature and that could lead to infinite values in the equation.
VSM: it has no requirement for smoothing.

In order to check the impact of the various smoothing possibilities we have devised 6 smoothing algorithms to be applied with the 3 ML methods (DL, NB, and VSM). We want to note that not requiring smoothing does

not mean that the method does not profit from the smoothing technique (as we shall see in the evaluation). For the baseline smoothing strategy we chose both "no smoothing", and "fixed smoothing"; we also tried a simple but competitive method from (Ng, 1997), denoted as "Ng smoothing" (methods to be described below). The other three possibilities consist on applying the Feature-Type method as in Figure V.2, with two variants: use "Ng smoothing" for back-off (E), or in a combined fashion (F):

(A) No smoothing: Use raw frequencies directly.
(B) Fixed smoothing: Assign 0.1 raw frequency to each sense with a 0 value.
(Ng) Ng smoothing: This method is based on the global distribution of the senses in the training data. For each feature, each of the senses of the target word that has no occurrences in the training data gets the ratio between the probability of the sense occurring in the training data and the total number of examples: *Prob(sense)/Number_of_examples*.
(Ft) Feature-type smoothing: The method described in this chapter. In the case of DL, note that when no data is available the feature is just discarded. For NB, it is necessary to rely on back-off strategies (see E and F).
(E) Ft with Ng as back-off: When Ft does not provide smoothed values, Ng is applied.
(F) Ft and Ng combined: The smoothed values are obtained by multiplying Ft and Ng values. Thus, in Figure V.2, the $count'(f, w, sense)$ values are multiplied by *Prob(sense)/Number_of_examples*.

The output of the smoothing algorithm is the list of counts that replace the original frequency counts when computing the probabilities. We tested all possible combinations, but notice that not all smoothing techniques can be used with all the methods (e.g. we cannot use NB with "no smoothing").

V.6.3 Application of smoothing: an example

We will focus on three feature types and the target word *art* in order to show how the smoothed values are computed. For *art*, the following features have a 1/0 distribution in the training data: *"prev_N_wf Aboriginal"*, *"win_cont_lem_context Jerry"*, and *"win_2gram_context collection owned"*[5].

[5]The first feature indicates that *Aboriginal* was the first noun to the left of *art*. The second that *Jerry* was found in the context window. The third that the bigram *collection*

X	Y	*prev_N_wf*				*win_cont_lem_context*				*win_2gram_context*			
		X'	Y'	X"	Y"	X'	Y'	X"	Y"	X'	Y'	X"	Y"
1	0	4	4	1	0.91	517	1187	1	2.24	63	150	1	2.31
2	0	6	1	2	1.18	82	125	2	4.45	8	4	2	4.37
3	0	2	0	3	1.14	13	22	3	6.62	2	1	3	6.48
...													

Table V.2: Smoothed values (interpolation per word) for the feature types *prev_N_wf*, *win_cont_lem_context* and *win_2gram_context* with the target word *art*.

The majority sense for the three cases is the first sense. If we find one of those features in a test occurrence of *art*, we would like to know whether they are good indicators of the first sense or not.

As all these features occur with frequency 1/0, we have collected all counts for the feature types (e.g. *prev_N_wf*) which also have 1/0 occurrences in the training data. Table V.1 shows the counts for *prev_N_wf*; the (4,4) values that appear for (X',Y') indicate that the *prev_N_wf* features that have 1/0 distribution in the target-folds contribute 4 examples to the majority sense and 4 to the minority senses when looked up in the estimation-folds.

The data for *prev_N_wf* has at least 3 points, and therefore we use the accumulated frequencies to obtain an interpolation table. We see that the interpolated frequencies for the minority senses stay nearly constant when the X values go up. This would reflect that the probability of the minority senses would go down quickly for higher values of X. In fact, the interpolated table can be used for values of X greater that 3, which had not been attested in the training data.

The same process is followed for the other two feature types: *win_cont_lem_context* and *win_2gram_context*. Table V.2 shows the smoothed values (X',Y') and the interpolated values (X",Y") for the three types studied. The values for Y are much higher in the latter two cases, indicating that there is a very low confidence for these features for the word *art*. In contrast, *prev_N_wf* can be a valuable feature if found in 4/0 or greater distributions.

Figure V.3 shows this different behavior graphically for *prev_N_wf* and *win_cont_lem_context*. For each feature type, the estimated Y" values and the log-ratio of the majority sense are given: the higher the Y" the lower

owned was found in the context window.

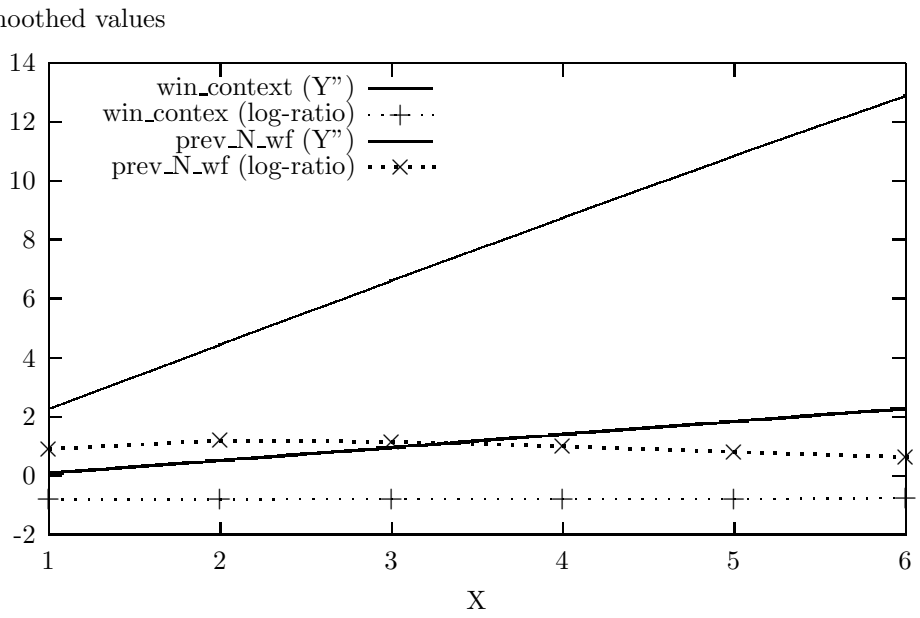

Figure V.3: Interpolation curves for the X/0 case (features *prev_N_wf* and *win_context*) with the target word *art*. The Y" estimation and the $log(X''/Y'')$ values are given for each X value and feature.

the confidence in the majority sense, and inversely for the log-ratio. We can see that the curve for the Y" values assigned to *prev_N_wf* get lower credit as X increases, and the log-ratio grows constantly. On the contrary, for *win_cont_lem_context* the values of Y" increase, and that the log-ratio remains below zero, indicating that this feature type is not informative.

V.7 Evaluation on Senseval-2

The main experiment is aimed at studying the performance of four ML methods with the different smoothing approaches (where applicable). The recall achieved on the Senseval-2 dataset is shown in table V.3, the best results per method marked in bold. We separated the results according to the type of smoothing: basic smoothing ("no smoothing" and "fixed smoothing"), and complex smoothing (techniques that rely on "Feature-type smoothing" and "Ng smoothing"). We can see that the results are different depending on the ML method, but the best results are achieved with complex smoothing for the 3 ML methods studied: DL (Ft and E), NB (F), and VSM (Ng). The

	Basic Smoothing		Complex Smoothing			
	A	**B**	**Ng**	**Ft**	**E**	**F**
DL		60.4	60.7	**64.4**	**64.4**	64.3
NB		62.9	63.5		61.8	**63.8**
VSM	65.9	65.6	**66.2**	64.0	64.2	65.2
SVM	**65.8**					

Table V.3: ML methods and smoothing techniques: (A) no smoothing, (B) fixed smoothing, (Ng) Ng smoothing, (Ft) Feature-type smoothing, the method presented in this chapter, (E) Ft with Ng as back-off, and (F) the combination of Ft and Ng.

best performance is attained by the VSM method, reaching 66.2%, which is one of the highest reported in this dataset. The other methods get more profit from the smoothing techniques, but their performance is clearly lower. McNemar's test[6] shows that the difference between the results of the best "basic smoothing" technique and the best "complex smoothing" technique is significant for DL and NB, but not for VSM.

All in all, we see that the performance of the statistically-based (DL, NB) methods improves significantly, making them comparable to the best single methods. In the next experiment, we tested a simple way to combine the output of the 4 systems: one system, one vote. The combination was tested on 2 types of systems: those that relied on "complex smoothing", and those that not. For each algorithm, the best smoothing technique for each type was chosen; e.g. the VSM algorithm would use the (A) approach for "simple smoothing", and (Ng) for "complex smoothing" (see table V.3). The performance of these systems is given in table V.4. The table also shows the results achieved discarding one system in turn.

The results show that we get an improvement over the best system (VSM) of 0.5% when combining it with DL and SVM. The table also illustrates that smoothing accounts for all the improvement, as the combination of methods with simple smoothing only reaches 66.0% in the best case, for 66.7% of the "complex smoothing" (difference statistically significant according to McNemar's test with 95% confidence interval).

As a reference, table V.5 shows the results reported for different groups

[6]McNemar's significance test has been applied with a 95% confidence interval.

Systems	Basic smoothing	Complex smoothing
All methods	65.7	66.2
except SVM	64.9	66.2
except NB	**66.0**	**66.7**
except VSM	64.9	65.7
except DL	65.7	66.3

Table V.4: Combination of systems with basic smoothing and complex smoothing. The rows show the recall achieved combining the 4 systems, and discarding one in turn.

and algorithms in the Senseval-2 competition and in more recent works. Our algorithms are identified by the "IXA" letters. "JHU - S2", corresponds to the Johns Hopkins University system in Senseval-2, which was the best performing system. "JHU" indicates the systems from the Johns Hopkins University implemented after Senseval-2 (Cucerzan and Yarowsky, 2003; Florian *et al.*, 2002). Finally, "NUS" (National University of Singapore) stands for the systems presented in (Lee and Ng, 2002). In addition to the methods that we applied; there are Mixture Models (MM), AdaBoost, and Decision Trees. The table is sorted by recall.

We can see that our systems achieve high performance, and that the combination of systems is able to beat the best results. However, we chose the best smoothing algorithm for the methods using the testing data (instead of using cross-validation on training, which would require to construct the smoothing tables for each fold). This fact makes the combined system not directly comparable. In any case, it seems clear that the system benefits from smoothing, and obtains results similar to the best figures reported to date.

V.8 Evaluation on Senseval-3

In this section we present the results obtained by our systems in two different tasks of the Senseval-3 competition (English and Basque lexical-sample tasks). An analysis of the supervised systems competing in the English task is given in section II.8. We used the augmented feature set described in section V.4, including domain features.

We applied the smoothing techniques and the ensemble of methods stud-

Method	Group	Smoothing	Recall	
Combination	IXA	Complex (best)	66.7	
Combination	**JHU**		**66.5**	⇒ **Best result to date**
VSM	IXA	Ng	66.2	
Combination	IXA	Basic (best)	66.0	
SVM	IXA		65.8	
SVM	**NUS**		**65.4**	⇒ **2nd best result to date**
DL	IXA	Ft	64.4	
Combination	**JHU-S2**		**64.2**	⇒ **Senseval-2 winner**
NB	IXA	E	63.8	
AdaBoost	NUS		62.7	
NB	NUS	"Add one"	62.7	
Mixture Models	JHU		62.5	
Decision Trees	NUS		57.2	

Table V.5: Comparison with the best systems in the Senseval-2 competition and the recent literature.

ied in this chapter. After evaluating the algorithms on the Senseval-3 training data by means of cross-validation, we submitted two systems for each task: the best ensemble, and the best single method.

Table V.6 shows the performance obtained by our systems and the winning systems (which are described in detail in section II.8) in the Senseval-3 evaluation. We can see that we are very close to the best algorithms in both languages.

Our best ensemble for English (in Senseval-3 training, by means of cross-validation) was formed by three systems: DL (with Ft smoothing), VSM (with Ng smoothing), and SVM. The combination of methods was useful for the final task, where we improve the recall of the best single system (VSM, with Ng smoothing) in 0.3%, reaching 72.3%. This difference is statistically significant according to McNemar's test. Our disambiguation procedure shows a similar behavior on the Senseval-2 and Senseval-3 data for English, where the ensemble works best, followed by VSM. The smoothing methods contribute to increase the recall in both cases.

However, the results are different for the Basque task. The cross-validation experiments indicate that the best combination is formed by the following systems: NB (Ng smoothing), VSM (Ng smoothing), and SVM. The best single system is SVM. In this case, the combination of methods does not im-

Task	System Code	Method	Recall
Eng.	**Senseval-3 Winner**	**RLSC - kernel**	**72,9**
Eng.	BCU_comb	DL(Ft)-VSM(Ng)-SVM	72,3
Eng.	BCU-English	VSM(Ng)	72,0
Basq.	**Senseval-3 Winner**	**AB**	**70,4**
Basq.	BCU-Basque	SVM	69,9
Basq.	BCU-Basque_comb	NB(Ng)-VSM(Ng)-SVM	69,5

Table V.6: Official results for the English and Basque lexical tasks (recall).

prove the results, and the SVM method alone provides better results (69.9% recall), although the difference is not significant applying McNemar's test. In general, the profit from the smoothing methods is much lower, and some algorithms (like VSM) seem to perform below the expectations. We think that the Basque feature set needs more analysis.

Overall, this task showed that the ensemble of algorithms (with the help of smoothing) provides a more robust system for WSD, and is able to achieve state-of-the-art performance.

V.9 Conclusions

In this work, we have studied the smoothing method proposed in (Yarowsky, 1995a), and we present a detailed algorithm for its application to WSD. We have described the parameters used, and we have applied the method on three different ML algorithms: DL, NB, and VSM. We also analyzed the impact of several smoothing strategies, and the combination of algorithms to construct a robust WSD system.

The evaluation on Senseval-2 data indicated that the smoothing method explored in this chapter is able to make all three methods perform at very high precisions, comparable and in some cases superior to the best result attained in the Senseval-2 competition. We also showed that a simple combination of the methods and a fourth system based on SVM attains the best result for the Senseval-2 competition reported so far (although only in its more successful configuration, as the system was not "frozen" using cross-validation). We also applied this architecture to the English and Basque lexical-sample tasks in Senseval-3. We submitted two systems for each task after tuning on cross-validation: the best ensemble, and the best single method. Our systems

obtained good results, very close to the winning systems in both tasks.

For English, our disambiguation method shows a similar behavior on the Senseval-2 and the Senseval-3 datasets (both in cross-validation and against the testing part). The ensemble performs best in all cases, followed by VSM. The smoothing methods contribute to increase the recall in both cases. VSM has proved to be a competitive single system, while the ensemble of algorithms provides robustness, achieving state-of-the-art performance.

The results for Basque are different, in this case the best single system is SVM, and the combination of methods does not improve the results. In general, the profit from the smoothing methods is much lower, and some algorithms (like VSM) seem to perform below the expectations. Our main conclusion for Basque is that the chosen feature set should be revised, as it is not clear how to represent the context in case of agglutinative languages. Using a "cleaner" feature set would also help the smoothing techniques. Another improvement for the Basque system would come from the inclusion of domain tags as features, using the information available in the Senseval-3 dataset[7].

For further study of the smoothing method, we would like to extend this work to X/Y features for Y greater than 1, and try other grouping criteria, e.g. taking into account the class of the word. We would also like to compare our results to other more general smoothing techniques (Good, 1953; Jelinek and Mercer, 1980; Chen, 1996).

An interesting application of the smoothing techniques is to detect good features, even in the case of low amounts of training data (as it is the case for most of the words in WSD). These features could be used as seeds to obtain new examples automatically, in a fashion similar to the method in (Leacock *et al.*, 1998), which will be studied and applied throughout the next chapter. They could also be integrated in a bootstrapping process using DLs, as in (Yarowsky, 1995b). The DL algorithm is well suited for this task, as it relies on a single piece of evidence (feature) to choose the correct sense, and it has been shown to perform significantly better with smoothing.

Finally, we would like to apply our last version of the algorithm, which has been shown to perform with state-of-the-art recall on the lexical sample, to an all-words task. The smoothing techniques would help us to address the sparse data problem; and the knowledge acquisition problem will be attacked

[7] A single experiment adding this simple feature to the best Basque system (VSM) showed an improvement of 0.6% recall, beating the Senseval-3 winner by 0.1%.

with the algorithm to be presented in the next chapter.

VI. CHAPTER

Automatic acquisition of sense-tagged examples

VI.1 Introduction

One of the main drawbacks for supervised WSD is the knowledge acquisition bottleneck: the systems need large amounts of costly hand-tagged data. The situation is more dramatic for lesser studied languages, like Basque. In order to overcome this problem, different research lines have been explored. The following are the most studied: bootstrapping techniques (Yarowsky, 1995b), active learning (Argamon-Engelson and Dagan, 1999), and automatic acquisition of training examples (Mihalcea, 2002). We will introduce each of these lines in the "related work" section. In this work, we have focused on the automatic acquisition of examples.

When supervised systems have no specific training examples for a target word, they need to rely on publicly available all-words sense-tagged corpora like Semcor, which is tagged with WordNet word senses. The systems performing best in the English all-words tasks in Senseval-2 and Senseval-3 (cf. chapter II) were basically supervised systems trained on Semcor and similar sources, like WordNet examples. Unfortunately, for most of the words, these corpora only provide a handful of tagged examples. In fact, only a few systems could overcome the MFS baseline[1] in the different editions of

[1] This value was obtained assigning the most frequent sense in Semcor.

the Senseval all-words task. In our approach, we will also rely on Semcor as the basic resource, both for training examples and as an indicator of the distribution of the senses of the target word.

The goal of this chapter is to evaluate up to which point we can automatically acquire examples for word senses and train accurate supervised WSD systems on them. This is a very promising line of research, but one which remains relatively under-studied (cf. section VI.2). The method we applied is based on the monosemous relatives of the target words (Leacock *et al.*, 1998), and we studied some parameters that affect the quality of the acquired corpus, such as the distribution of the number of training instances per each word sense (bias), and the type of features used for disambiguation (local vs. topical).

Basically, we built three systems with different degrees of supervision that would be applicable to an all-words task:

- Fully supervised: using examples from Semcor and automatically acquired examples.
- Minimally supervised: using the distribution of senses in Semcor and automatically acquired examples.
- Fully unsupervised: using an automatically acquired sense rank (McCarthy *et al.*, 2004) and automatically acquired examples.

This chapter is structured as follows. First, section VI.2 describes previous work on the field. section VI.3 introduces the experimental setting for evaluating the acquired corpus, and section VI.4 presents the feature set. section VI.5 is devoted to the process of building the corpus, which is evaluated in section VI.6. Finally, the conclusions are given in section VI.7.

VI.2 Related work

As we mentioned in the introduction, three main lines of research on the knowledge acquisition bottleneck are bootstrapping techniques, active learning, and automatic acquisition of training examples. We will briefly introduce the former two, and then we will focus on related work on automatic acquisition of examples, which is the goal of this chapter.

Bootstrapping techniques consist on algorithms that learn from labeled and unlabeled data. Among them, we can highlight co-training (Blum and Mitchell, 1998) and their derivatives (Collins and Singer, 1999; Abney, 2002). These techniques are very appropriate for WSD and other NLP tasks because of the wide availability of untagged data and the scarcity of tagged data. However, there is no published positive results for WSD. In his well-known work, Yarowsky (1995b) applied an iterative bootstrapping process to induce a classifier based on DLs. With a minimum set of seed (annotated) examples, disambiguation results comparable to supervised methods were obtained in a limited set of binary sense distinctions, but this work has not been extended to fine-grained senses.

Active learning is used to choose informative examples for hand-tagging, in order to reduce the manual cost. Argamon-Engelson and Dagan (1999) describe two main types of active learning: membership queries and selective sampling. In the first approach, the learner constructs examples and asks a teacher to label them. This approach would be difficult to apply to WSD. Instead, in selective sampling the learner selects the most informative examples from unlabeled data. In one of the few works directly applied to WSD, Fujii *et al.* (1998) applied selective sampling to the acquisition of examples for disambiguation of verb senses, in an iterative process with human taggers. The informative examples were chosen following two criteria: maximum number of neighbors in unsupervised data, and minimum similarity with the supervised example set. Another active learning approach is the Open Mind Word Expert (Mihalcea and Chklovski, 2003), which is a project to collect sense-tagged examples from web users. The system selects the examples to be tagged applying a selective sampling method based on two different classifiers, choosing the unlabeled examples where there is disagreement. The collected data was used in the Senseval-3 English lexical-sample task (cf. section II.3).

In automatic acquisition of training examples, an external lexical source (WordNet, for instance) or a sense-tagged corpus is used to obtain new examples from a very large untagged corpus (e.g. the Internet). In (Leacock *et al.*, 1998), a method to obtain sense-tagged examples using monosemous relatives from WordNet is presented. Our approach, which will be described in section VI.5, is based on this early work. In their algorithm, Leacock *et al.* (1998) retrieve the same number of examples per each sense, and they give preference to monosemous relatives that consist in a multiword containing the target word. Their experiment is evaluated on 3 words (a noun, a

verb, and an adjective) with coarse sense-granularity and few senses. The results showed that the monosemous corpus provided precision comparable to hand-tagged data.

In another approach, Mihalcea and Moldovan (1999) used information in WordNet (e.g. monosemous synonyms and glosses) to construct queries, which were later fed into the Altavista[2] web search engine. Four procedures were used sequentially, in a decreasing order of precision, but with increasing levels of coverage. Results were evaluated by hand, finding out that over 91% of the examples were correctly retrieved among a set of 1,080 instances of 120 word senses. However, the number of examples acquired did not have to correlate with the frequency of senses, and the corpus resulting from the experiment was not used for training a real WSD system.

In a related work, Mihalcea (2002) generated a sense tagged corpus (GenCor) by using a set of seeds consisting of sense-tagged examples from four sources: SemCor, WordNet, examples created using the method above, and hand-tagged examples from other sources (e.g. the Senseval-2 corpus). By means of an iterative process, the system obtained new seeds from the retrieved examples. In total, a corpus with about 160,000 examples was gathered. The evaluation in the lexical-sample task showed that the method was useful for a subset of the Senseval-2 testing words (results for 5 words were provided).

VI.3 Experimental Setting

For the experiments in this chapter we chose the "Senseval2B" setting (cf. section III.3.3.4). In this setting, the examples on the Senseval-2 testing data tagged with multiwords, phrasal verbs, and proper nouns are previously removed in order to focus on the sense distinctions of each word.

The experiments were performed on the 29 nouns available for the Senseval-2 lexical-sample task. We separated these nouns in 2 sets, depending on the number of examples they have in Semcor: Set A contained the 16 nouns with more than 10 examples in Semcor, and Set B the remaining low-frequency words.

It is important to note that the training part of Senseval-2 lexical-sample was not used in the process, as our goal was to test the performance we could

[2] http://www.altavista.com

achieve with the minimal resources (i.e. those available for any word).

VI.4 Feature set

As features, we relied on a basic set of local and topical features. In previous chapters we have seen that richer features can improve the performance of the system, but in this case we focused on the comparison of hand-tagged and automatically-obtained corpora, and therefore the overall performance of the systems was not relevant.

Previous work on automatic acquisition of examples (Leacock *et al.*, 1998) has reported lower performance when using local collocations formed by PoS tags or closed-class words. We analyzed the results using local and topical features separately, and also the combination of both types:

- Local features: Bigrams and trigrams, formed by the word-form, lemma, and part-of-speech of the surrounding words. Also the content lemmas in a ± 4 word window around the target.
- Topical features: All the content lemmas in the context.

VI.5 Building the monosemous relatives web corpus

In order to build this corpus[3], we have acquired 1,000 Google snippets for each monosemous word in WordNet 1.7. Then, for each word sense of the ambiguous words, we gathered the examples of its monosemous relatives (see below). This method is inspired in (Leacock *et al.*, 1998), and has shown to be effective in experiments of topic signature acquisition (Agirre and Lopez de Lacalle, 2004). This last paper also shows that it is possible to gather examples based on monosemous relatives for nearly all noun senses in WordNet[4].

The basic assumption is that for a given word sense of the target word, if we had a monosemous synonym of the word sense, then the examples of the synonym should be very similar to the target word sense, and could

[3]The automatically acquired corpus will be referred indistinctly as web-corpus, or monosemous-corpus

[4]All the examples in this work are publicly available in `http://ixa2.si.ehu.es/pub/sensecorpus`

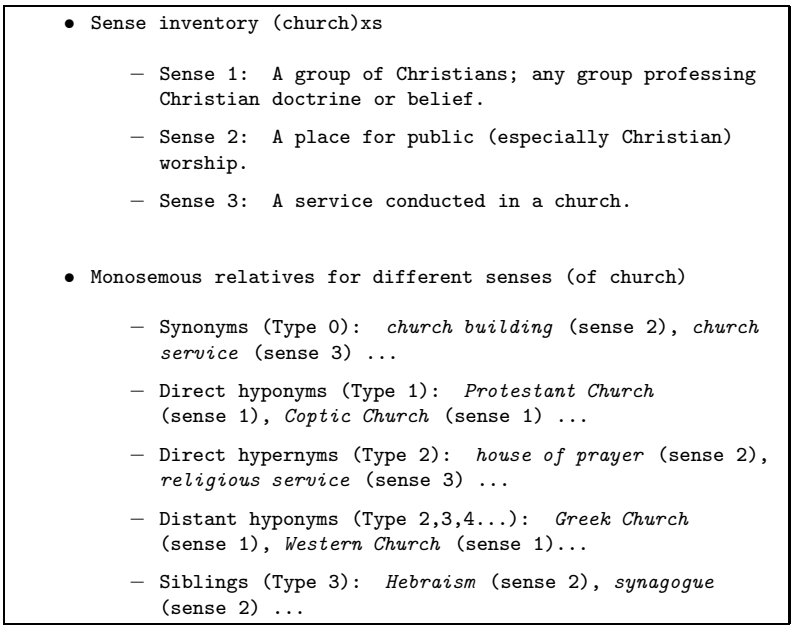

Figure VI.1: Sense inventory and a sample of monosemous relatives in WordNet 1.7 for *church*.

therefore be used to train a classifier of the target word sense. The same, but in a lesser extent, can be applied to other monosemous relatives, such as direct hyponyms, direct hypernyms, siblings, indirect hyponyms, etc. The expected reliability decreases with the distance in the hierarchy from the monosemous relative to the target word sense.

The monosemous-corpus was built using the simplest technique: we collected examples from the web for each of the monosemous relatives. The relatives have an associated number (type), which correlates roughly with the distance to the target word, and indicates their relevance: the higher the type, the less reliable the relative. A sample of monosemous relatives for different senses of *church*, together with its sense inventory in WordNet 1.7 is shown in figure VI.1.

Distant hyponyms receive a type number equal to the distance to the target sense. Note that we assigned a higher type value to direct hypernyms than to direct hyponyms, as the latter are more useful for disambiguation. We also decided to include siblings, but with a high type value.

In the following subsections we will describe step by step the method to

construct the corpus. First we will explain the acquisition of the highest possible amount of examples per sense; then we will explain different ways to limit the number of examples per sense for a better performance; finally we will see the effect of training on local or topical features on this kind of corpora.

VI.5.1 Collecting the examples

The examples are collected following these steps:

1: We query Google[5] with the monosemous relatives for each sense, and we extract the snippets as returned by the search engine. All snippets returned by Google are used (up to 1,000). The list of snippets is sorted in reverse order. This is done because the top hits usually are titles and incomplete sentences that are not useful.

2: We extract the sentences (or fragments of sentences) around the target search term. Some of the sentences are discarded, according to the following criteria: length shorter than 6 words, having more non-alphanumeric characters than words divided by two, or having more words in uppercase than in lowercase.

3: The automatically acquired examples contain a monosemous relative of the target word. In order to use these examples to train the classifiers, the monosemous relative (which can be a multiword term) is substituted by the target word. In the case of the monosemous relative being a multiword that contains the target word (e.g. *Protestant Church* for *church*) we can choose not to substitute, because *Protestant*, for instance, can be a useful feature for the first sense of church. In these cases, we decided not to substitute and keep the original sentence, as our preliminary experiments on this corpus suggested (although the differences were not significant).

4: For a given word sense, we collect the desired number of examples (see following section) in order of type: we first retrieve all examples of type 0, then type 1, etc. up to type 3 until the necessary examples are obtained. We did not collect examples from type 4 upwards. We did not make any distinctions between the relatives from each type. Leacock *et al.* (1998) give preference to multiword relatives containing the target word, which could be an improvement in future work.

[5]We use the offline XML interface kindly provided by Google for research.

On average, we have acquired roughly 24,000 examples for each of the target words used in this experiment.

VI.5.2 Number of examples per sense (bias)

Previous work (Agirre and Martinez, 2000) has reported that the distribution of the number of examples per word sense (bias for short) has a strong influence in the quality of the results. That is, the results degrade significantly whenever the training and testing samples have different distributions of the senses.

As we are extracting examples automatically, we have to decide how many examples we will use for each sense. In order to test the impact of bias, different settings have been tried:

- No bias: we take an equal amount of examples for each sense.
- Web bias: we take all examples gathered from the web.
- Automatic ranking: the number of examples is given by a ranking obtained following the method described in (McCarthy *et al.*, 2004). They used a thesaurus automatically created from the BNC corpus with the method from (Lin, 1998a), coupled with WordNet-based similarity measures.
- Semcor bias: we take a number of examples proportional to the bias of the word senses in Semcor.

For example, table VI.1 shows the number of examples per type (0,1,...) that are acquired for *church* following the Semcor bias. The last column gives the number of examples in Semcor.

We have to note that the three first methods do not require any hand-labeled data, and that the fourth relies in Semcor.

The way to apply the bias is not straightforward in some cases. In our first approach for Semcor-bias, we assigned 1,000 examples to the major sense in Semcor, and gave the other senses their proportion of examples (when available). But in some cases the distribution of the Semcor bias and that of the actual examples in the web would not fit. The problem is caused when there are not enough examples in the web to fill the expectations of a certain word sense.

We therefore tried another distribution. We computed, for each word, the minimum ratio of examples that were available for a given target bias

Sense	0	1	2	3	Total	Semcor
church#1	0	476	524	0	1,000	60
church#2	306	100	561	0	967	58
church#3	147	0	20	0	167	10
Overall	453	576	1,105	0	2,134	128

Table VI.1: Examples per type (0,1,...) that are acquired from the web for the three senses of *church* following the Semcor bias, and total number of examples in Semcor.

Sense	Semcor		Web corpus								Senseval test	
			Web bias		Semcor Pr		Semcor MR		Auto. MR			
	#ex	%	#ex	%	#ex	%	#ex	%	#ex	%	#ex	%
authority#1	18	60	338	0.5	338	33.7	324	59.9	138	19.3	37	37.4
authority#2	5	16.7	44932	66.4	277	27.6	90	16.6	75	10.5	17	17.2
authority#3	3	10	10798	16	166	16.6	54	10.0	93	13.0	1	1.0
authority#4	2	6.7	886	1.3	111	11.1	36	6.7	67	9.4	0	0
authority#5	1	3.3	6526	9.6	55	5.5	18	3.3	205	28.6	34	34.3
authority#6	1	3.3	71	0.1	55	5.5	18	3.3	71	9.9	10	10.1
authority#7	0	0	4106	6.1	1	0.1	1	0.2	67	9.4	0	0
Overall	30	100	67657	100	1003	100	541	100	716	100	99	100

Table VI.2: Distribution of examples for the senses of *authority* in different corpora. Pr (proportional) and MR (minimum ratio) columns correspond to different ways to apply Semcor bias.

Word	Web bias	Semcor bias	Automatic bias
art	15,387	10,656	2,610
authority	67,657	541	716
bar	50,925	16,627	5,329
bum	17,244	2,555	4,745
chair	24,625	8,512	2,111
channel	31,582	3,235	10,015
child	47,619	3,504	791
church	8,704	5,376	6,355
circuit	21,977	3,588	5,095
day	84,448	9,690	3,660
detention	2,650	1,510	511
dyke	4,210	1,367	843
facility	11,049	8,578	1,196
fatigue	6,237	3,438	5,477
feeling	9,601	1,160	945
grip	20,874	2,209	277
hearth	6,682	1,531	2,730
holiday	16,714	1,248	1,846
lady	12,161	2,959	884
material	100,109	7,855	6,385
mouth	648	287	464
nation	608	594	608
nature	32,553	24,746	9,813
post	34,968	4,264	8,005
restraint	33,055	2,152	2,877
sense	10,315	2,059	2,176
spade	5,361	2,458	2,657
stress	10,356	2,175	3,081
yew	10,767	2,000	8,013
Average	24,137	4,719	3,455
Total	699,086	136,874	100,215

Table VI.3: Number of examples following different sense distributions for the Senseval-2 nouns. Minimum-ratio is applied for the Semcor and automatic bias.

and a given number of examples extracted from the web. We observed that this last approach would reflect better the original bias, at the cost of having less examples.

Table VI.2 presents the different distributions of examples for *authority*. There we can see the Senseval-testing and Semcor distributions, together with the total number of examples in the web; the Semcor proportional distribution (Pr) and minimum ratio (MR); and the automatic distribution (MR). The table illustrates how the proportional Semcor bias produces a corpus where the percentage of some of the senses is different from that in Semcor, e.g. the first sense only gets 33.7% of the examples, in contrast to the 60% it had in Semcor.

We can also see how the distributions of senses in Semcor and Senseval-test have important differences, although the main sense is the same. For the web and automatic distributions, the first sense is different; and in the case of the web distribution, the first hand-tagged sense only accounts for 0.5% of the examples retrieved from the web. Similar distribution discrepancies can be observed for most of the words in the test set. The *Semcor MR* column shows how using minimum ratio we get a better reflection of the proportion of examples in Semcor, compared to the simpler proportional approach (*Semcor Pr*) . For the automatic bias we only used the minimum ratio.

To conclude this section, table VI.3 shows the number of examples acquired automatically for each word following three approaches: the web bias, the Semcor bias with minimum ratio, and the Automatic bias with minimum ratio. We can see that retrieving all the examples we get 24,137 examples in average per word; and respectively 4,700 or 3,400 if we apply the Semcor bias or the Automatic bias.

VI.5.3 Local vs. topical features

Previous work on automatic acquisition of examples (Leacock *et al.*, 1998) has reported lower performance when using local collocations formed by PoS tags or closed-class words. We performed an early experiment comparing the results using local features, topical features, and a combination of both. In this case we used the web corpus with Senseval training bias, distributed according to the MR approach, and always substituting the target word. The recall (per word and overall) is given in table VI.4.

In this setting, we observed that local collocations achieved the best precision overall, but the combination of all features obtained the best recall. Local features achieve 58.5% precision for 96.7% coverage overall, while topical and combination of features have full-coverage.

There were clear differences in the results per word, suggesting that estimating the best feature-set per word would improve the performance. For the evaluation experiments, we chose to work with the combination of all features.

Word	Local Feats.			Topical Feats.	Combination
	Coverage	Precision	Recall	Recall	Recall
art	94.4	57.4	54.2	45.6	47.0
authority	93.4	51.2	47.8	43.2	46.2
bar	98.3	53.0	52.1	55.9	57.2
bum	100	81.2	81.2	87.5	85.0
chair	100	88.7	88.7	88.7	88.7
channel	73.5	54.0	39.7	53.7	55.9
child	100	56.5	56.5	55.6	56.5
church	100	67.7	67.7	51.6	54.8
circuit	88.7	51.1	45.3	54.2	56.1
day	98.6	60.2	59.4	54.7	56.8
detention	100	87.5	87.5	87.5	87.5
dyke	100	89.3	89.3	89.3	89.3
facility	98.2	29.1	28.6	21.4	21.4
fatigue	100	82.5	82.5	82.5	82.5
feeling	100	55.1	55.1	60.2	60.2
grip	100	19.0	19.0	38.0	39.0
hearth	100	73.4	73.4	75.0	75.0
holiday	100	96.3	96.3	96.3	96.3
lady	100	80.4	80.4	73.9	73.9
material	100	43.2	43.2	44.2	43.8
mouth	100	36.8	36.8	38.6	39.5
nation	100	80.6	80.6	80.6	80.6
nature	100	44.4	44.4	39.3	40.7
post	98.3	44.7	43.9	40.5	40.5
restraint	79.5	37.1	29.5	37.5	37.1
sense	93.0	62.5	58.1	37.2	38.4
spade	100	74.2	74.2	72.6	74.2
stress	100	53.9	53.9	46.1	48.7
yew	100	81.5	81.5	81.5	81.5
Overall	96.7	58.5	56.5	56.0	57.0

Table VI.4: Results per feature type (local, topical, and combination), using the monosemous corpus with Senseval-2 training bias (MR, and substitution). Coverage and precision are given only for local features (topical and combination have full coverage).

VI.6 Evaluation

In all experiments, the recall of the systems is presented as evaluation measure. There is total coverage (because of the high overlap of topical features) and the recall and precision are the same.

In order to evaluate the acquired corpus, our first task was to analyze the impact of bias. The overall results are shown in table VI.5. There are 2 figures for each distribution: obtained simply assigning the first ranked sense (1st sense), and using the monosemous corpus following the predetermined

Bias	Type	1st sense	Train exam.	Diff.
no bias		18.3	38.0	+19.7
web bias	unsuperv.	33.3	39.8	+6.5
autom. ranking		36.1	43.2	+7.1
Semcor bias	minimally-	47.8	49.8	+2.0
Senseval2 bias	supervised	55.6	57.5	+1.9

Table VI.5: Performance (recall) on the Senseval-2 lexical-sample, using different biases to create the corpus. The *type* column shows the kind of system.

bias (Train exam.). As we described in section VI.3, the testing part of the Senseval-2 lexical sample data was used for evaluation. We also include the results using Senseval2 bias, which is taken from the training part. The recall per word for some distributions can be seen in table VI.4.

The results show clearly that when bias information from a hand-tagged corpora is used the recall improves significantly, even when the bias comes from a corpus -Semcor- different from the target corpus -Senseval-. The bias is useful by itself, and we see that the higher the performance of the 1st ranked sense heuristic, the lower the gain using the monosemous corpus. We want to note that in fully unsupervised mode we attain a recall of 43.2% with the automatic ranking. Using the minimally supervised information of bias, we get 49.8% if we have the bias from an external corpus (Semcor) and 57.5% if we have access to the bias of the target corpus (Senseval[6]). This results show clearly that the acquired corpus has useful information about the word senses, and that bias is extremely important.

The results per word are given in table VI.6. We can see that if we do not use some kind of sense-distributional information the results for some words drop below 10% precision using web bias: *child*, *day*, *grip*, ...

We will present two further experiments performed with the monosemous corpus resource. The goal of the first will be to measure the WSD performance that we achieve using Semcor as the only supervised data source. In our second experiment, we will compare the performance of our totally unsupervised approach (monosemous corpus and automatic bias) with other unsupervised approaches in the Senseval-2 English lexical task.

[6]Bias obtained from the training-set.

Word	Unsupervised		Supervised		
	No bias	Web bias	Autom. ranking	Semcor bias	Senseval2 bias
art	34.0	61.1	45.6	55.6	44.9
authority	20.9	22.0	40.0	41.8	46.2
bar	24.7	52.1	26.4	51.6	57.2
bum	36.7	18.8	57.5	5.0	85.0
chair	61.3	62.9	69.4	88.7	88.7
channel	42.2	28.7	30.9	16.2	57.4
child	40.3	1.6	34.7	54.0	58.9
church	43.8	62.1	49.7	48.4	51.6
circuit	44.3	52.8	49.1	41.5	58.0
day	15.3	2.2	12.5	48.0	60.4
detention	52.1	16.7	87.5	52.1	87.5
dyke	92.9	89.3	80.4	92.9	89.3
facility	19.6	26.8	22.0	26.8	21.4
fatigue	58.8	73.8	75.0	82.5	82.5
feeling	27.2	51.0	42.5	60.2	60.2
grip	11.3	8.0	28.2	16.0	38.0
hearth	57.8	37.5	60.4	75.0	75.0
holiday	70.4	7.4	72.2	96.3	96.3
lady	24.3	79.3	23.9	80.4	73.9
material	51.7	50.8	52.3	54.2	42.9
mouth	39.5	39.5	46.5	54.4	39.5
nation	80.6	80.6	80.6	80.6	80.6
nature	21.9	44.4	34.1	46.7	40.7
post	36.8	47.4	47.4	34.2	40.5
restraint	26.3	9.1	31.4	27.3	37.1
sense	44.8	18.6	41.9	47.7	48.8
spade	74.2	66.1	85.5	67.7	74.2
stress	38.6	52.6	27.6	2.6	48.7
yew	70.4	85.2	77.8	66.7	81.5
Overall	38.0	39.8	43.2	49.8	57.5

Table VI.6: Performance (recall) on the Senseval-2 lexical-sample per word, using different biases to create the corpus.

VI.6.1 Monosemous corpus and Semcor bias

In this experiment we compared the performance using the monosemous corpus (with Semcor bias and minimum ratio), and the examples from Semcor. We noted that there were clear differences depending on the number of training examples for each word, therefore we studied each word-set described in section VI.3 separately. The results per word-set are shown in table VI.7. The figures correspond to the recall training in Semcor, the web-corpus, and the combination of both.

If we focus on set B (words with less than 10 examples in Semcor), we see that the MFS figure is very low (40.1%). There are some words that do not have any occurrence in Semcor, and thus the sense is chosen at random.

Word-set	**MFS**	**Semcor**	**Web**	**Semcor + Web**	**MFS & Web**
set A (> 10)	**51.9**	50.5	50.9	51.6	**51.9**
set B (< 10)	40.1	-	47.7	**47.8**	**47.8**
all words	47.8	47.4	49.8	50.3	**50.5**

Table VI.7: Recall training in Semcor, the acquired web corpus (Semcor bias), and a combination of both, compared to that of the Semcor MFS.

It made no sense to train the DL for this set, therefore this result is not in the table. For this set, the bias information from Semcor is also scarce, but the DLs trained on the web-corpus raise the performance to 47.8%.

For set A, the average number of examples is higher, and this raises the results for Semcor MFS (51.9%). We see that the recall for DL training in Semcor is lower that the MFS baseline (50.5%). The main reasons for these low results are the differences between the training and testing corpora (Semcor and Senseval). There have been previous works on portability of hand-tagged corpora that show how some constraints, like the genre or topic of the corpus, affect heavily the results (Martinez and Agirre, 2000). If we train on the web-corpus the results improve, and the best results are obtained with the combination of both corpora, reaching 51.6%. We need to note, however, that this is still lower than the Semcor MFS.

Finally, we will examine the results for the whole set of nouns in the Senseval-2 lexical-sample (last row in table VI.7), where we see that the best approach relies on the web-corpus. In order to disambiguate the 29 nouns using only Semcor, we apply MFS when there are less than 10 examples (set B), and train the DLs for the rest.

The results in table VI.7 show that the web-corpus raises recall, and the best results are obtained combining the Semcor data and the web examples (50.3%). As we noted, the web-corpus is specially useful when there are few examples in Semcor (set B), therefore we made another test, using the web-corpus only for set B, and applying MFS for set A. The recall was slightly better (50.5%), as is shown in the last column.

VI.6.2 Monosemous corpus and Automatic bias (unsupervised method)

In this experiment we compared the performance of our unsupervised system with other approaches. For this goal, we used the resources available from the Senseval-2 competition, where the answers of the participating systems in the different tasks were available[7]. This made possible to compare our results and those of other systems deemed unsupervised by the organizers on the same test data and set of nouns.

From the 5 unsupervised systems presented in the Senseval-2 lexical-sample task as unsupervised, the *WASP-Bench* system relied on lexicographers to hand-code information semi-automatically (Tugwell and Kilgarriff, 2001). This system does not use the training data, but as it uses manually coded knowledge we think it falls clearly in the supervised category.

The results for the other 4 systems and our own are shown in table VI.8. We show the results for the totally unsupervised system and the minimally unsupervised system (Semcor bias). We classified the *UNED* system (Fernandez-Amoros *et al.*, 2001) as minimally supervised. It does not use hand-tagged examples for training, but some of the heuristics that are applied by the system rely on the bias information available in Semcor. The distribution of senses is used to discard low-frequency senses, and also to choose the first sense as a back-off strategy. On the same conditions, our minimally supervised system attains 49.8 recall, nearly 5 points more.

The rest of the systems are fully unsupervised, and they perform significantly worse than our system.

VI.7 Conclusions

This chapter explores the large-scale acquisition of sense-tagged examples for WSD, which is a very promising line of research, but remains relatively under-studied. We have applied the "monosemous relatives" method to construct automatically a web corpus which we have used to train three systems based on DL: one fully supervised (applying examples from Semcor and the web corpus), one minimally supervised (relying on the distribution of senses in Semcor and the web corpus) and another fully unsupervised (using an

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

Method	Type	Recall
Web corpus (Semcor bias)	minimally-	**49.8**
UNED	supervised	45.1
Web corpus (Autom. bias)		**43.3**
Kenneth_Litkowski-clr-ls	unsupervised	35.8
Haynes-IIT2		27.9
Haynes-IIT1		26.4

Table VI.8: Our minimally supervised and fully unsupervised systems compared to the unsupervised systems (marked in bold) in the 29 noun subset of the Senseval-2 Lexical Sample.

automatically acquired sense rank and the web corpus). Those systems were tested on the Senseval-2 lexical sample test set.

We have shown that the fully supervised system combining our web corpus with the examples in Semcor improves over the same system trained on Semcor alone. This improvement is specially noticeable in the nouns that have less than 10 examples in Semcor. Regarding the minimally supervised and fully unsupervised systems, we have shown that they perform well better than the other systems of the same category presented in the Senseval-2 lexical-sample competition. The system can be trained for all nouns in WordNet, using the data collected from the web, and it is publicly available[8].

The research also highlights the importance of bias. Knowing how many examples are to be fed into the ML system is a key issue. We have explored several possibilities, and we have seen that assigning directly the first sense in a ranking obtained from hand-tagged data (or even with automatic means on raw corpora) can be a good approximation for disambiguation. However, the DL algorithm is always able to improve this heuristic training on the automatically acquired examples.

We think that this research opens the opportunity for further improvements. We have to note that the MFS heuristic and the supervised systems based on the Senseval-2 training data are well ahead of our results, and our research aims at investigating ideas to close this gap. Some experiments in the line of adding automatically retrieved examples to available hand-tagged data (Semcor and Senseval-2) have been explored. The preliminary results indicate that this process has to be performed carefully, taking into account

[8] `http://ixa2.si.ehu.es/pub/sensecorpus`

the bias of the senses and applying a quality-check of the examples before they are included in the training data.

In order to improve the system, in the future we would like to apply more powerful ML methods, like the ensemble constructed on chapter V. We would also like to tune the algorithm that chooses the monosemous relatives, giving preference, for instance, to multiwords that contain the target word as in (Leacock *et al.*, 1998). The method could also benefit from sophisticated tools to acquire examples that are now available, like ExRetriever (Fernandez *et al.*, 2004), which could open the way to examples with less noise and better performance. Another idea to enrich the system would be to retrieve examples by queries based on collocations. These collocations would have a strong relation with specific senses, and could be detected with the smoothing techniques described in chapter V.

We would also like to apply this method for new languages and testbeds. An interesting approach could be to retrieve examples for languages that count on lexical databases like WordNet, but do not have an all-words sense-tagged corpora (e.g. Basque). Moreover, now that the monosemous corpus is available for all nouns in English, we would like to test the system on the all-words task, analyzing specially words with low amounts of available hand-tagged data.

Finally, we want to note that our results suggest that there is a portability problem when extending hand-tagged corpora with new examples (see also (Escudero *et al.*, 2000c)). In this chapter, we have addressed the problem by means of sense-rankings, obtained from hand-tagged data and automatically. However, in order to construct a robust system, we should take into account how the change of corpora and domain affects WSD performance. We will address this issue in the next chapter, relying on the DSO corpus, which contains examples from two different corpora (BC and WSJ).

VII. CHAPTER

Portability and genre/topic of corpora

VII.1 Introduction

The previous chapter has presented us the difficulties of extending available hand-tagged data by automatic means. We emphasized the importance of the bias of the sense distribution, and we showed that, without some prior information, the performance drops dramatically when training on automatically-acquired examples. However, another factor that has to be taken into account when we add new examples to a corpus is portability. In previous work, the application of WSD systems trained on a given corpus to be tested on a different one has shown a drop in performance, even when applying tuning techniques (Escudero *et al.*, 2000c). One of the drawbacks of using different corpora, highlighted by Ng *et al.* (1999), is that when the hand-tagging of the same corpus is performed by independent teams of researchers there is low inter-tagger agreement. Another important issue is the fact that new examples come usually from different genre and topics.

In order to be able to alleviate the knowledge acquisition bottleneck and extend our corpora, we have to study the reasons for this degradation of performance. In the early nineties, two famous papers claimed that the behavior of word senses in texts adhered to two principles: one sense per discourse (Gale *et al.*, 1993) and one sense per collocation (Yarowsky, 1993). The first constraint states that words the occurrences of a word tend to have the same meaning in a given discourse. The "one sense per collocation" rule

mantains that the collocations in the nearby context contain strong clues that serve to determin the meaning of a word. These principles (specially the second) have been widely used to construct supervised WSD systems. The hypotheses were shown to hold for some particular corpora (totaling 380 Mwords) on words with 2-way ambiguity. The word sense distinctions came from different sources (translations into French, homophones, homographs, pseudo-words, etc.), but no dictionary or lexical resource was linked to them. In the case of the "one sense per collocation" paper, several corpora were used, but no study was done to show whether the collocations hold across corpora. We think that revisiting these hypotheses in the present framework of supervised WSD (with fine-grained sense distinctions and new resources), could provide us insight on the portability of WSD systems.

Krovetz (1998) showed that the "one sense per discourse" hypothesis does not hold for fine-grained senses in SemCor and DSO, as 33% of the words in these corpora had multiple meanings in the same discourse. His results have been confirmed in our own experiments. We will therefore concentrate on the "one sense per collocation" hypothesis, considering these two questions:

1. Does the collocation hypothesis hold for fine-grained sense distinctions (compared to homograph level granularity)?

2. Does the collocation hypothesis hold across corpora, that is, across genre and topic variations (compared to a single corpus, probably with little genre and topic variations)?

In order to try to answer the above questions, we will rely on the DSO collection (cf. section II.3), which comprises texts from two different corpora: BC and WSJ. We will first compare the strength of the "one sense per collocation" hypothesis in cross-corpora tagging with the figures obtained using one single corpus. Then we will measure the effect of the discourse (deriving training and testing examples from the same documents) on the results. Finally, we will test the influence of the genre and topic of the examples in WSD performance across different sections of the BC, which cover different genres and topics. We think that this study will highlight the factors that come into play when porting a WSD system to a new corpus, and help us to build more robust WSD algorithms.

This chapter is organized as follows. Section VII.2 described related work on portability of WSD systems. The resources used and the experimental settings are presented in section VII.3. Section VII.4 presents the collocations

considered. Sections VII.5 and VII.6 show the in-corpus and cross-corpora experiments, respectively. Section VII.7 discusses the effect of drawing training and testing data from the same documents. Section VII.8 evaluates the impact of genre and topic variations, which is further discussed in section VII.9. Finally, section VII.10 presents some conclusions.

VII.2 Related work

The portability of large-scale hand-tagged corpora was first analyzed in the work by Ng *et al.* (1999). This work relied on the overlapping examples in the DSO and Semcor corpora, which belonged to the BC corpus. They measured the tagger agreement for the two teams that developed DSO and Semcor. They report low values both in precision (56.7%) and Kappa coefficient[1] (0.317). They also present an algorithm that builds coarser senses from the human annotations, and they suggest that this inventory can be used to better evaluate WSD algorithms.

Escudero *et al.* (2000c) perform a set of experiments on cross-corpora tagging using four different ML methods (including AB and NB). They test the portability of the systems using the two parts of DSO: WSJ and BC. These corpora is combined in several ways; for instance, training on the whole DSO and testing on BC. They apply cross-validation when training and testing parts overlap. The results reported in cross-corpora tagging are low, in some cases below the MFS baseline. In another experiment from this article, they tested a tuning method that consisted on including for training some examples from the target corpus. The goal was to test whether having some examples from the target corpus would be enough to profit from a different corpus. They analyzed the learning curve, adding more examples from the target at each step. The results were not good, as AB was the only ML method that profited slightly from the different corpus.

In their analysis, Escudero *et al.* (2000c) studied the differences in the sense distributions in WSJ and BC, but they did not take into account the different genre and topic of the documents. While in the WSJ corpus all the texts come from press articles, the BC is balanced, with some sections belonging to the "press" domain (cf. section II.3). We used this information

[1] The Kappa coefficient measures the agreement between annotators after factoring out the effect of chance agreement. A value of 0 indicates that the agreement is purely due to chance, while the maximum value of 1 indicates full agreement.

for our work on this chapter, in order to study the effect of the domain on portability.

In (McCarthy *et al.*, 2004), they implement a method to obtain automatically a ranking of senses for the corpus they want to disambiguate. The algorithm, which we described briefly in section VI.5.2, proceeds building a thesaurus with the method from (Lin, 1998a), and applying WordNet-based similarity measures. As it is shown in their experiments, the algorithm is able to acquire information of the preferred senses for different domains. They note that the first sense heuristic is used by many WSD systems as back-off strategy (specially for all-words tasks), and it is also applied for lexical acquisition. We already applied this technique in chapter VI, in order to obtain bias information for the automatic acquisition of examples from the web. As we showed, using their ranking method and the examples from the web, we were able to build a totally unsupervised WSD system that outperformed other systems in the Senseval-2 English lexical-sample task. We think that the technique from (McCarthy *et al.*, 2004) offers promising results for the portability of WSD systems.

VII.3 Experimental setting

The experiments in this chapter were performed using the WSJ&BC setting (section III.3.3.2), which consisted on the DSO corpora, and the C word-set (21 nouns and verbs). The two sources of the DSO corpus (WSJ and BC) are used separately for cross-tagging experiments.

As mentioned earlier, the WSJ contains press articles, and the BC is balanced, with the texts classified according some predefined categories (as we can see in section II.3). These categories have been previously used in genre detection experiments (Karlgren and Cutting, 1994), where each category was used as a genre. We think that the categories not only reflect genre variations but also topic variations (e.g. the *Religion* category follows topic distinctions rather than genre). Nevertheless we are aware that some topics can be covered in more than one category. Unfortunately we could not find a topically tagged corpus which also have word sense tags. We thus speak of genre and topic variation, knowing that further analysis would be needed to measure the effect of each of them.

As usual, we use 10-fold cross-validation when training and testing on the same corpus. When comparing the performance on decision lists trained on

two different corpora (or sub-corpora) we always take an equal amount of examples per word from each corpora. This is done to discard the amount-of-data factor.

VII.4 Feature set

In order to test the "one sense per collocation" rule presented in (Yarowsky, 1993), we will adopt the broad definition of collocations used in that work, which corresponds to the basic feature types we are using throughout this dissertation. Therefore, from now on we will speak indistinctly of collocations and features, although the term "collocations" is often applied to refer to non-compositional, high-frequency word co-occurrences (Firth, 1957). If a more strict linguistic perspective is taken, rather than collocations we should speak about co-occurrence relations.

The collocations that we studied were classified into three subsets: local content word collocations, local part-of-speech and function-word collocations, and global content-word collocations. The "local content word" subset is the only one that would adhere to the narrower definition of collocation. We only considered those collocations that could be easily extracted from a part of speech tagged corpus, like "word to left", "word to right", etc. Local content word collocations comprise bigrams ("word to left", "word to right") and trigrams ("two words to left", "two words to right", and "both words to right and left"). At least one of those words needs to be a content word. Local function-word collocations comprise also all kinds of bigrams and trigrams, as before, but the words need to be function words. Local PoS collocations take the Part of Speech of the words in the bigrams and trigrams. Finally, global content word collocations comprise the content words around the target word in two different contexts: a window of ± 4 words around the target word, and all the words in the sentence. Table VII.1 summarizes the collocations used. These collocations have been used in other word sense disambiguation research and are also referred to as features (Gale *et al.*, 1993; Ng and Lee, 1996; Escudero *et al.*, 2000c).

Compared to (Yarowsky, 1993), who also took into account grammatical relations, we only share the content-word-to-left and the content-word-to-right collocations. We did not lemmatize content words, and we therefore do take into account the form of the target word. For instance, *governing body* and *governing bodies* are different collocations for the sake of this chapter.

Local content word collocations	
Word-to-left	Content Word
Word-to-right	Content Word
Two-words-to-left	At least one Content Word
Two-words-to-right	
Word-to-right-and-left	
Local PoS and function word collocations	
Word-to-left	Function Word&PoS
Word-to-right	Function Word&PoS
Two-words-to-left	Both Function Words&PoS
Two-words-to-right	PoS
Word-to-right-and-left	PoS
Global content word collocations	
Word in Window of ±4w	Content Word
Word in sentence	

Table VII.1: Types of collocations.

VII.5 In-corpus experiments

We extracted the collocations in the BC section of the DSO corpus and, using 10-fold cross-validation, tagged the same corpus. The same procedure was followed for the WSJ part. The precision and coverage results are shown in tables VII.2 and VII.3, where the collocation groups are given in bold. We can observe the following:

- The best kinds of collocations are local content word collocations, especially if two words from the context are taken into consideration, but the coverage is low. Function words to right and left also attain remarkable precision.

- Collocations are stronger in the WSJ, surely due to the fact that the BC is balanced, and therefore includes more genres and topics. This is a first indicator that genre and topic variations have to be taken into account.

- Collocations for fine-grained word-senses are sensibly weaker than those reported by Yarowsky (1993) for two-way ambiguous words. Yarowsky reports 99% precision, while our highest results do not reach 80%.

Collocations	Nouns		Verbs		Overall	
	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.
Word-to-right	64.4	20.3	43.2	23.0	56.2	21.2
Word-to-left	62.6	12.4	77.0	13.9	68.1	12.9
Two-words-to-right	65.7	14.6	50.0	10.3	61.3	13.1
Two-words-to-left	74.0	9.2	81.9	12.2	77.4	10.3
Word-to-right-and-left	64.7	8.8	68.6	11.4	66.3	9.8
Overall local content	**67.5**	**40.5**	**63.5**	**40.4**	**66.1**	**40.5**
Word-to-right	48.0	50.3	45.2	40.6	47.1	46.8
Word-to-left	41.4	63.9	57.2	52.7	46.4	59.9
Two-words-to-right	52.0	18.3	62.4	11.3	54.7	15.8
Two-words-to-left	42.0	13.1	64.8	17.3	51.6	14.6
Word-to-right-and-left	54.9	23.8	65.4	16.0	57.7	21.0
PoS-to-right	34.0	99.2	35.6	99.2	34.6	99.2
PoS -to-left	35.0	99.4	48.3	99.2	39.8	99.3
Two- PoS -to-right	40.6	92.3	42.2	87.6	41.2	90.6
Two- PoS -to-left	39.6	79.2	53.9	89.7	45.2	82.9
PoS -to-right-and-left	41.6	92.1	54.5	88.5	46.1	90.8
Overall local PoS&Fun	**48.6**	**100**	**56.0**	**100**	**51.2**	**100**
Word in sentence	54.5	100,0	49.2	100,0	52.6	100,0
Word in Window of 4	55.0	97.2	52.5	95.1	54.1	96.4
Overall global content	**54.9**	**100**	**50.3**	**100**	**53.3**	**100**
OVERALL	**57.7**	**100**	**56.4**	**100**	**57.2**	**100**

Table VII.2: Train on BC, tag BC.

Collocations	Nouns		Verbs		Overall	
	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.
Word-to-right	76.8	25.4	52.9	26.4	68.0	25.8
Word-to-left	72.4	18.5	86.7	18.2	77.5	18.4
Two-words-to-right	78.4	19.1	62.3	11.3	74.4	16.3
Two-words-to-left	81.1	16.0	86.2	17.9	83.0	16.6
Word-to-right-and-left	82.0	16.9	72.8	12.9	79.3	15.5
Overall local content	**76.4**	**50.2**	**73.7**	**49.7**	**75.5**	**50.0**
Word-to-right	60.0	45.7	52.7	37.0	57.7	42.6
Word-to-left	54.5	60.9	62.9	47.2	57.0	56.0
Two-words-to-right	63.8	13.3	68.7	8.4	65.0	11.6
Two-words-to-left	60.0	14.0	65.7	10.8	61.7	12.8
Word-to-right-and-left	72.1	22.0	69.4	13.8	71.4	19.1
PoS-to-right	49.0	99.3	48.8	99.3	48.9	99.3
PoS -to-left	46.5	99.1	58.4	99.4	50.8	99.2
Two- PoS -to-right	52.6	91.8	53.4	87.9	52.9	90.4
Two- PoS -to-left	51.8	82.2	61.4	91.2	55.5	85.4
PoS -to-right-and-left	55.5	91.8	63.4	89.1	58.3	90.8
Overall local PoS&Fun	**62.2**	**100**	**64.0**	**100**	**62.9**	**100**
Word in sentence	61.1	100	57.2	100	59.7	100
Word in Window of 4	62.7	97.9	61.1	97.5	62.2	97.7
Overall global content	**61.7**	**100**	**58.0**	**100**	**60.4**	**100**
OVERALL	**66.1**	**100**	**63.5**	**100**	**65.2**	**100**

Table VII.3: Train on WSJ, tag WSJ.

It has to be noted that the test and training examples come from the same corpus, which means that, for some test cases, there are training examples from the same document. In some sense we can say that one sense per discourse comes into play. This point will be further explored in section VII.7.

In the rest of this chapter, only the overall results for each subset of the collocations will be shown. We will pay special attention to local-content collocations, as they are the strongest, and also closer to strict definitions of collocation.

As an example of the learned collocations, table VII.4 shows some strong local content word collocations for the noun *state*, and figure VII.1 shows the word senses of *state* (6 out of the 8 senses are shown as the rest were not present in the corpora).

```
1. state -- (the group of people comprising the government of a
   sovereign state)

2. state, province -- (the territory occupied by one of the
   constituent administrative districts of a nation)

3. state, nation, country, land, commonwealth, res publica, body
   politic -- (a politically organized body of people under a
   single government)

4. state -- (the way something is with respect to its main
   attributes)

5. Department of State, State Department, State -- (the federal
   department that sets and maintains foreign policies)

6. country, state, land, nation -- (the territory occupied by a
   nation)
```

Figure VII.1: Word senses for *state* in WordNet 1.6 (6 out of 8 are shown).

VII.6 Cross-corpora experiments

In these experiments we train on the BC and tag the WSJ corpus and vice versa. Tables VII.5 and VII.6, when compared to tables VII.2 and VII.3 show a significant drop in performance (both precision and coverage) for all kinds of collocations (we only show the results for each subset of collocations). For instance, table VII.5 shows a drop in 16% in precision for local content collocations when compared to table VII.3.

Collocation	Log	Sense of *state*					
		#1	#2	#3	#4	#5	#6
State government	3.68	-	-	-	-	4	-
six states	3.68	-	-	-	-	4	-
State 's largest	3.68	-	-	-	-	4	-
State of emergency	3.68	-	4	-	-	-	-
Federal , state	3.68	-	-	-	-	4	-
State , including	3.68	-	-	-	-	4	-
Current state of	3.40	-	3	-	-	-	-
State aid	3.40	-	-	-	3	-	-
State where Farmers	3.40	3	-	-	-	-	-
State of mind	3.40	-	3	-	-	-	-
Current state	3.40	-	3	-	-	-	-
State thrift	3.40	-	-	-	3	-	-
Distributable state aid	3.40	-	-	-	3	-	-
State judges	3.40	-	-	-	-	3	-
a state court	3.40	-	-	3	-	-	-
said the state	3.40	-	-	-	-	3	-
Several states	3.40	-	-	-	-	3	-
State monopolies	3.40	-	-	-	3	-	-
State laws	3.40	-	-	3	-	-	-
State aid bonds	3.40	-	-	-	3	-	-
Distributable state	3.40	-	-	-	3	-	-
State and local	2.01	-	-	1	1	15	-
Federal and state	1.60	-	-	-	1	5	-
State court	1.38	-	-	12	-	3	-
Other state .	1.38	4	-	-	-	1	-
State governments	1.09	1	-	-	-	3	-

Table VII.4: Local content-word collocations for *state* in the WSJ. For each sense and collocation, the number of examples is given.

These results confirm those by Escudero *et al.* (2000c) who conclude that the information learned in one corpus is not useful by itself to tag the other.

In order to analyze the reason for this performance degradation, we compared the local content word collocations extracted from one corpus and the other. Table VII.7 shows the amount of collocations extracted from each corpus, how many of the collocations are shared on average and how many of the shared collocations are in contradiction. The low amount of collocations shared between both corpora could explain the poor figures, but for some words (e.g. *point*) there is a worrying proportion of contradicting collocations.

We inspected some of the contradicting collocations and saw that in all the cases they were caused by errors (or at least differing criteria) of the hand-taggers when dealing with words with difficult sense distinctions. For instance, table VII.8 shows some collocations of *point* which receive contra-

Collocations	Nouns		Verbs		Overall	
	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.
Overall local content	59.7	33.8	59.1	35.6	59.5	34.4
Overall local PoS&Fun	47.8	99.9	49.1	99.7	48.3	99.8
Overall global content	44.2	100	45.5	99.9	44.7	100
OVERALL	48.5	100	49.7	100	48.9	100

Table VII.5: Cross-corpora tagging: train on BC, tag WSJ.

Collocations	Nouns		Verbs		Overall	
	Pr.	Cov.	Pr.	Cov.	Pr.	Cov.
Overall local content	51.2	27.3	55.6	33.6	53	29.5
Overall local PoS&Fun	42.1	100	48.6	100	44.4	100
Overall global content	39.2	100	42.3	100	40.3	100
OVERALL	42.9	100	48.3	100	44.8	100

Table VII.6: Cross-corpora tagging: train on WSJ, tag BC.

dictory senses in the BC and the WSJ. The collocation *point of view*, for instance, is assigned the fourth sense in 13 out of 15 occurrences in the BC, and the second sense in all 19 occurrences in the WSJ.

We can therefore conclude that the "one sense per collocation" holds across corpora, because the contradictions found were due to tagging errors. The low amount of collocations in common would explain by itself the low figures on cross-corpora tagging.

But yet, we wanted to further study the reasons for the low number of collocations in common, which affects the cross-corpora performance. We thought of different factors that could come into play:

a. Effect of discourse: the training and test examples from the in-corpus experiments are taken at random, and they could be drawn from the same document. This could make the results appear better for in-corpora experiments. On the contrary, in the cross-corpora experiments, training and testing examples come always from different documents.

b. Genre and topic changes caused by the shift from one corpus to the other.

Word	PoS	Coll. BC	Coll. WSJ	% Coll. Shared	% Coll. Contradict.
Age	N	45	60	27	0
Art	N	24	35	34	20
Body	N	12	20	12	0
Car	N	92	99	17	0
Child	N	77	111	40	05
Cost	N	88	88	32	0
Head	N	77	95	07	33
Interest	N	80	141	32	33
Line	N	110	145	20	38
Point	N	44	44	32	86
State	N	196	214	28	48
Thing	N	197	183	66	52
Work	N	112	149	46	63
Become	V	182	225	51	15
Fall	V	36	68	19	60
Grow	V	61	71	36	33
Lose	V	63	56	47	43
Set	V	94	113	54	43
Speak	V	34	38	28	0
Strike	V	12	17	14	0
Tell	V	137	190	45	57

Table VII.7: Collocations shared and in contradiction between BC and WSJ.

Collocation	BC			WSJ		
	#2	#4	Others	#2	#4	Others
important point	3	0	0	0	2	0
point of view	1	13	1	19	0	0

Table VII.8: Contradictory senses of *point* in BC and WSJ.

c. Corpora have intrinsic features that cannot be captured by sole genre and topic variations.

d. The size of the data, being small, would account for the low amount of collocations shared.

We explore the factor a) in section VII.7, and b) in section VII.8. The latter points c) and d) are discussed in section VII.9.

VII.7 Effect of discourse

In order to test whether drawing training and testing examples from the same document explains the different performance in in-corpora and cross-corpora

tagging, we performed the following experiment. Instead of organizing the 10 random subsets for cross-validation on the examples, we chose 10 subsets of the documents (also at random). This way, the testing examples and training examples are guaranteed to come from different documents. We also think that this experiment would show more realistic performance figures, as a real application should not expect to find examples from the documents used for training.

Unfortunately, there are not any explicit document boundaries, neither in the BC nor in the WSJ. In the BC, we took files as documents, even if files might contain more than one excerpt from different documents. This procedure guarantees that document boundaries are not crossed. It has to be noted that following this organization the target examples would share fewer examples from the same topic. The 168 files from the BC were divided in 10 subsets at random: we took 8 subsets with 17 files and 2 subsets with 16 files.

For the WSJ, the only cue was the directory organization. In this case we were unsure about the meaning of this organization, but hand inspection showed that document boundaries were not crossing discourse boundaries. The 61 directories were divided in 9 subsets with 6 directories and 1 subset with 7.

Again, 10-fold cross-validation was used on these subsets and the results in tables VII.9 and VII.10 were obtained. The tables show the results per each part of speech using the main collocation groups: all the features (overall), and the local content features. The *Diff.* column shows the change in precision with respect to tables VII.2 and VII.3, which separate the folds according to examples instead of documents.

Table VII.9 shows that for the BC, precision and coverage are degraded significantly, compared to table VII.2. On the contrary, the results for the WSJ are nearly the same (cf. tables VII.10 and VII.3).

The results for WSJ indicate that drawing training and testing data from the same or different documents in itself does not affect so much the results. The degradation of the BC results could be explained by the greater variation in topic and genre. This factor will be further studied in section VII.8.

Finally, table VII.11 summarizes the overall results on WSJ and BC for each of the different experiments performed. The figures show that drawing training and testing data from the same or different documents would not in any case explain the low figures in cross-corpora tagging.

PoS	Overall			Local content		
	Pr.	Cov.	Diff.	Pr.	Cov.	Diff.
N	49.9	100	-7.8	57.3	30.7	-10.2
V	54.3	100	-2.1	60.8	37.9	-2.7
Overall	51.4	100	-5.8	58.7	33.3	-7.4

Table VII.9: Train on BC, tag BC, cross-validation according to files.

PoS	Overall			Local content		
	Pr.	Cov.	Diff.	Pr.	Cov.	Diff.
N	65.0	100	-1.1	76.2	48.6	-0.2
V	63.4	100	-0.1	69.7	49.4	-4.0
Overall	64.4	100	-1.1	73.8	48.9	-1.7

Table VII.10: Train on WSJ, tag WSJ, cross-validation according to files.

Testing Corpus	In-corpora(examples)	In-corpora(files)	Cross-corpora
BC	57.2	51.4	44.8
WSJ	65.2	64.4	48.9

Table VII.11: Overall results in different experiments tagging WSJ and BC (overall features).

VII.8 Effect of genre/topic variations

Trying to shed some light on this issue, we observed that the category *press:reportage* in BNC, is related to the genre/topic of the WSJ. We therefore designed the following experiment: we tagged each category in the BC with the DLs trained on the rest of the categories in the BC, and also with the DLs trained on the WSJ.

Tables VII.12 and VII.13 show respectively the results for the local content collocation and the overall collocation set. As the latter set has full coverage, only precision figures are given. For local content-word collocations, table VII.12 illustrates that training on the WSJ attains the best precision and coverage for *press:reportage*, both compared to the results for the other categories, and to the results reached by the rest of the BC on *press:reportage*.

Therefore we can say that:

1. From all the categories, the local content collocations from *press: reportage* are the most similar to those of WSJ.

2. WSJ contains collocations which are closer to those of *press: reportage*, than those from the rest of the BC.

As for table VII.13, we see that introducing the whole set of collocations (with weaker collocations that have higher coverage) shows some changes in the results. In this case, the better performing category when training on WSJ is not *press:reportage*, although it is between the best. However, we see that this category is the only to improve (together with *Miscellaneous*) when trained on WSJ, as happened with local features.

In this table, we can see that the inclusion of weak features increases the overlapping between the collocations for all categories, at the cost of being less reliable. This factor will normally decrease the precision, although for some categories these features work surprisingly well (*Mystery*, *Humor*, ...). The effect of the genre/topic is less noticeable in these conditions, however, the results show again that the *press:reportage* category in the BC is more related to the WSJ corpus than to the rest of BC. In other words, having related genre/topic helps having common collocations, and therefore better WSD performance.

VII.9 Discussion

The goal of sections VII.7 and VII.8 was to explore the possible causes for the low number of collocations in common between BC and WSJ. Section VII.7 concludes that drawing the examples from different files is not the main reason for the degradation. This is specially true when the corpus has low genre/topic variation (e.g. WSJ). Section VII.8 shows that sharing genre/topic is a key factor, as the WSJ corpus attains better results on the *press:reportage* category than the rest of the categories on the BC itself. Texts on the same genre/topic share more collocations than texts on disparate genre/topics, even if they come from different corpora.

This seems to also rule out explanation c) in section VII.6 (corpora have intrinsic features unattainable from different sources), as a good measure of topic/genre similarity would help overcome cross-corpora problems.

That only leaves the low amount of data available for this study (explanation d). It is true that data scarcity can affect the number of collocations

Testing BC Category	Train on WSJ		Train on rest of BC	
	prec.	cov.	prec.	cov.
Press: Reportage	**62.5**	33.0	54.1	28.5
Skills and Hobbies	56.9	29.6	**57.1**	30.2
Romance and ...	56.1	27.1	**59.5**	34.0
Adventure and ...	55.1	22.3	**70.2**	31.2
Miscellaneous	**53.4**	32.1	**53.4**	30.4
General Fiction	52.5	23.9	**60.5**	32.1
Mystery and ...	52.3	24.3	**61.8**	36.9
Learned	51.8	25.7	**56.3**	28.0
Humor	51.6	32.1	**52.4**	33.7
Belles Lettres, ...	51.6	27.2	**52.4**	31.4
Press: Editorial	50.4	28.3	**59.3**	33.4
Popular Lore	48.8	30.4	**56.3**	35.3
Science Fiction	45.9	21.1	**58.6**	30.7
Press: Reviews	43.8	26.8	**48.8**	40.4
Religion	40.9	30.6	**53.7**	32.6

Table VII.12: Tagging different categories in BC (local content features). Results sorted by precision training on WSJ. Best precision results are shown in bold.

Testing BC Category	Train on WSJ	Train on rest of BC
Mystery and ...	55.1	**66.4**
Humor	52.0	**57.0**
Adventure and ...	49.7	**62.8**
Press: Reportage	**48.3**	45.7
Miscellaneous	**47.5**	40.2
Romance and ...	47.0	**56.5**
Popular Lore	46.4	**50.7**
General Fiction	44.6	**57.5**
Press: Editorial	44.1	**46.4**
Learned	43.8	**46.1**
Skills and Hobbies	43.5	**46.8**
Science Fiction	41.0	**56.5**
Belles Lettres, ...	40.9	**47.8**
Press: Reviews	40.9	**42.9**
Religion	37.2	**45.8**

Table VII.13: Precision tagging different categories in BC (overall features). Results sorted by precision training on WSJ. Best precision results are shown in bold.

shared across corpora. We think that larger data will make this number grow, especially if the corpus draws texts from different genres and topics. Nevertheless, the figures in table VII.12 indicate that even in these conditions genre/topic relatedness would help to find common collocations.

VII.10 Conclusions

This chapter shows that the one sense per collocation hypothesis is weaker for fine-grained word sense distinctions (e.g. those in WordNet): from the 99% precision mentioned for 2-way ambiguities in (Yarowsky, 1993) we drop to 70% figures. These figures could perhaps be improved using more available data.

We also show that one sense per collocation does hold across corpora, but that collocations vary from one corpus to other, following genre and topic variations. This explains the low results when performing word sense disambiguation across corpora. In fact, we demonstrated that when two independent corpora share a related genre/topic, the WSD results are better.

This has considerable impact in future work on WSD, as genre and topic are shown to be crucial parameters. A system trained on a specific genre/topic would have difficulties to adapt to new genre/topics. Besides, methods that try to extend automatically the amount of examples for training need also to account for genre and topic variations. We think that techniques like the one presented in (McCarthy *et al.*, 2004) can help to adapt WSD systems to new domains.

As a side effect, we have shown that the results on cross-validation WSD exercises, which mix training and test data drawn from the same documents, are higher than those from a more realistic setting. We also discovered several hand-tagging errors, which distorted extracted collocations. We did not evaluate the extent of these errors, but they certainly affected the performance on cross-corpora tagging.

In order to extend this work, one of the factors that should be analyzed is the separate influence of the genre and topic variations. The behaviour of different words through different corpora should also be addressed. Finally, ways to integrate genre/topic parameters into the WSD models have to be devised in order to build a general WSD system, as well as for lexical acquisition methods.

VIII. CHAPTER

Conclusions

We started this dissertation motivating a WSD tool in order to get more insight in our way to automatic natural language understanding. As we pointed out in the introduction, there are several complex factors that have to be addressed for this task. We tried to enumerate the main issues involving this problem in the introduction chapter, and the dissertation has been organized around these points:

1. The description of the problem: Explicit WSD with a closed list of senses may not be a correct way to model the intermediate task required for NLP.

2. The selection of a sense inventory: It is important that the senses of the words are represented with a good level of generalization, in order to be useful for NLP applications.

3. The application of ML algorithms: When choosing the methods that are being adapted from the ML community, the peculiarities of the WSD problem have to be taken into account.

4. The feature sets used to model the language: In order to be robust, the ML methods should rely in as much information from the texts as possible. Features obtained with complex analysis of the text (morphological, syntactic, semantic, domain, etc.) and the combination of different types of features could be used.

5. The sparse data problem: In NLP most of the events occur rarely, even when large quantities of training data are available. This problem is specially noticeable in WSD, where hand-tagged data is difficult to obtain.

6. Need for extra training data: Existing hand-tagged corpora do not seem enough for current state-of-the-art WSD systems. Hand-tagged data is difficult and costly to obtain. Estimations of the required tagging effort are not optimistic, and methods to obtain data automatically should be explored.

7. *Portability.* The porting of the WSD systems to be tested on a different corpora than the one used for training also presents difficulties. Previous work (Ng *et al.*, 1999; Escudero *et al.*, 2000c) has shown that there is a loss of performance when training on one corpora and testing on another. This has happened with automatically-tagged corpora, and also with corpora hand-tagged by independent teams of researchers.

In this dissertation, we have focused on the points 3–7, and the two first issues are out of the scope of our research. As we said in the introduction, there are other ways to approach the lexical ambiguity problem (see for instance (Kilgarriff and Tugwell, 2002)), but we explored the supervised approach to explicit WSD.

Thus, the points 3–7 comprise the main contents of the dissertation: baseline WSD system, feature types, smoothing, acquisition of examples, and portability. This book has been organized as follows: the issues 3 and 5 in the list (ML methods and sparse data problem) were addressed in chapter V; and each of the other issues was covered in a separate chapter with the same number they have in the list.

From the experiments on each of the aspects of supervised WSD, we were able to extract some conclusions. We will try first to summarize what we consider the main contributions of this dissertation, and then, in a more focalized way, we will describe the conclusions derived from each of the studied issues.

VIII.1 Contributions of our work

We will now describe the main contributions of this work, which were already advanced in chapter I. We will first present the **main research results**, and then we will introduce the tools and resources that have been built during this work:

- **Syntactic features:** In chapter IV, we explored the contribution of an extensive set of syntactic features to WSD performance. The experiments showed that basic and syntactic features contain complementary information, and that their integration is useful for WSD. The study included two different ML methods (DL and AB), and a precision/coverage trade-off system using different feature types. The contribution of syntactic features is specially noticeable for the algorithm AB in the standard setting, and for DLs when applying the precision/coverage trade-off.

- **Semantic features:** Also in chapter IV, we applied two approaches to study the contribution of semantic features using the WordNet hierarchy and the Semcor all-words corpus. On the one hand, we constructed new feature types based on the synsets surrounding the target word, the hypernyms of these synsets (at different levels), and also their semantic files. On the other hand, we learned different models of selectional preferences for verbs, using the relations extracted from the Semcor corpus by Minipar. Our main conclusions were that the "bag-of-synsets" approach that we applied does not improve the results; however, selectional preference acquisition offers promising results with a view to their integration with other feature types.

- **Automatic acquisition of examples:** In chapter VI, we applied a method to automatically acquire tagged examples from the web. This method, based on (Leacock *et al.*, 1998), obtained good performance on three systems with different supervision requirements: fully supervised (automatic examples added to hand-tagged corpora), minimally supervised (requiring information about sense distributions), and unsupervised (without hand-tagged examples). We showed that the fully supervised system, combining our web corpus with the examples in Semcor, improves over the same system trained on Semcor alone (spe-

cially for nouns with few examples in Semcor). Regarding the minimally supervised and fully unsupervised systems, we demonstrated that they perform well better than the other systems of the same category presented in the Senseval-2 lexical-sample competition. Our system can be trained for all nouns in WordNet, using the data collected from the web.

- **Genre/topic shift:** In chapter VII, we studied the strength of the "one sense per collocation" hypothesis (Yarowsky, 1993) using different corpora for training and testing. Our experiments show that the hypothesis is weaker for fine-grained word sense distinctions, and that it does hold across corpora, but that collocations vary from one corpus to other, following genre and topic variations. This would explain the low performance for WSD across corpora. In fact, we showed that when two independent corpora share a related genre/topic, the WSD results are better. Thus, this factor should be taken into account when extending the training data.

Other interesting results that came out from our work are the following **tools**:

- **High-precision WSD tool for English (chapter IV)**: We tested on Senseval-2 data different systems that could provide high precision at the cost of coverage. The results were promising, as two methods based on DLs reached 93% precision at 7% coverage (decision-threshold method), and 86% precision at 26% coverage (feature selection method). Syntactic features are specially helpful for feature selection.

- **Supervised WSD tool for English (chapter V)**: We developed a supervised system based on the combination of different ML methods and smoothing techniques. In the Senseval-3 English lexical-sample task, it ranked 5th among 47 submissions, only 0.6% lower than the best system. This system also participated in the all-words task, as a component of the "Meaning" system, which ranked 5th among 26 systems.

- **Supervised WSD tool for Basque (chapter V)**: We have adapted our models to Basque, which is an agglutinative language and presents

new challenges when defining the feature set. We have tested this tool on the Senseval-3 Basque lexical-sample task data, and it outperforms the results of other systems that took part in the event.

- **Unsupervised WSD tool for English (chapter VI)**: We built an unsupervised system relying on automatically obtained examples, which shows promising results for alleviating the knowledge acquisition bottleneck. It has been tested on the Senseval-2 English lexical-sample task, presenting the best performance of this kind of systems.

There are also some **lexical resources** (available for research) that have been developed as a result of our work:

- **Selectional preferences (chapter IV)**: Using the syntactic dependencies (object and subject) extracted from Semcor, we constructed and evaluated selectional preferences for verb and noun classes in WordNet. This database, consisting on weighted relations between synsets, is available by means of a Meaning license, or by personal request.

- **Sense tagged corpus (chapter VI)**: We constructed automatically a sense-tagged corpus for all nouns in WordNet. This resource is publicly available, and can be downloaded from `http://ixa2.si.ehu.es/pub/sensecorpus`.

Finally, during this research, we have published our results in different articles. The complete list is given in appendix A.

VIII.2 Detailed conclusions

In order to achieve the results described in the previous section, we followed a path through different WSD issues, which served to organize the chapters of this dissertation. The conclusions derived from our analysis were presented at the end of each chapter. We will now devote this section to summarize the main results.

Baseline WSD system under different conditions (3rd chapter)

These are our conclusions from our study on DL, "classic" features, and currently available hand-tagged data on different conditions:

- **Performance:** Semcor provides enough data to perform some basic general disambiguation, at 68% precision on any general running text. The performance on different words is similar, as ambiguity and number of examples are balanced in this corpus. The main differences are given by the PoS of the target words: the verbs present the highest polisemy and lowest precision. DSO provides large amounts of data for specific words, allowing for improved precision. However, it is unable to overcome the 70% barrier for our target word set. Finally, when applied to the Senseval-2 dataset, the system gets lower performance, with a recall of around 57% for the lexical-sample and all-words tasks. The main reasons for the low results are the high ambiguity of the target word-set (for the lexical-sample task), and the unavailability of training data (for the all-words task).

- **Relation between polisemy/bias/frequency and performance:** The highest results can be expected for words with a dominating word sense, but the difference to the MFS baseline is lower. Words with high polisemy tend to be the most frequent, which makes the polisemy and frequency factors balance each other.

- **Local features vs. topical features:** In Semcor, topical features were better for nouns, but not for other categories. For DSO, the local features achieved better performance than the topical set for all categories. This could be due to the much higher number of examples in DSO. It is important to note that single words exhibit different behavior, suggesting that the best policy could be the construction of word-experts with specific feature sets (Hoste *et al.*, 2002).

- **Learning curve:** The learning curve in Semcor shows that more data would help to improve the WSD system. In DSO, the system keeps learning with more data, but it stabilizes with 80% of all the available data, which indicates that a plateau has been reached for this system with 930 examples per noun and 1,370 examples per verb.

- **Noise in the data:** Our conclusion was that when we have few examples to train, as in Semcor, the noise affects the performance heavily, and it is necessary to use bigger amounts of data in order to minimize the damage.

- **Coarse-grained disambiguation:** The precision we obtain with semantic files is 83%, both in DSO and Semcor; but with slightly lower coverage in Semcor. The improvement is specially noticeable for verbs, where the reduction of sense granularity allows to reach 91% recall in DSO. An open issue is to find applications where coarse disambiguation would help.

- **Performance for Basque:** Our aim was to imitate the expressiveness of the well-studied features for English WSD, and we introduced several different feature types with that goal. However, a better study of the contribution of single features should be done. All in all, the results in the Senseval-2 task are encouraging, with our system only 2% below the winner JHU system (while the difference was 8% between these systems for English), which would indicate that our feature set represented better the context than the JHU set, although their ML method was clearly better.

Richer feature sets (4th chapter):

The types of features we have analyzed in this chapter are divided in three groups: syntactic features, semantic features, and selectional preferences. These were the main conclusions of our experiments:

- **Syntactic features on Semcor and DSO:** the performance in this setting was low, specially in Semcor (both precision and coverage). For DSO, the precision was slightly better than the basic set, but the coverage was low. The syntactic features did not contribute significantly in combination. Another study was conducted separately for the different feature types, and we observed that some syntactic features achieved comparatively good recall for verbs, specially ngrams, suggesting that some subcategorization information had been acquired. For further analysis, we focused on some words in the Semcor experiment, and analyzed the learned decision lists. These are the main problems observed:

  - Low coverage because of the sparse data.

  - Redundancy with basic types.

  - Presence of noisy features.
  - Parsing errors.

- **Syntactic features on the Senseval-2 corpus**

  The results were significantly better in this corpus for syntactic features. Syntactic features alone obtained better F1 value than the MFS baseline. The F1 value was much higher in this experiment than in the DSO task, even when the recall of the MFS baseline was higher in DSO. In our next experiment, we tested the combination of basic and syntactic features using the two ML methods. We extracted these conclusions:

  - AB outperforms DL in all cases, except for local features.
  - Syntactic features get worse results than local features.
  - Syntactic features prove to be useful in the combination. DLs profit from the additional syntactic features but the difference is only statistically significant for verbs. AB is able to attain significant improvement (1.8% overall, 2.7% for verbs).

- **Syntactic features and high precision systems**

  We analyzed two systems based on DL (Feature selection and Decision-threshold), and one based on AB (Decision-threshold). These are the main observations:

  - Syntactic features always help to improve the F1 of the basic set.
  - Adjusting the methods to a minimum loss of coverage (discarding the most difficult testing examples), the overall F1 improves for the three methods.
  - The methods based on DL reach 93% precision at 7% coverage (decision-threshold), and 86% precision at 26% coverage (feature selection). Syntactic features are specially helpful for feature selection.

  - AB does not achieve high precision figures, but it obtains the highest F1 score in this setting, with 66.7% precision and 84.5% coverage.

- **Semantic features:**

  This feature set was defined using the WordNet hierarchy, and the information from the semantic files. The experiments were performed on Semcor, which means that there were few examples to train, but also that the system would be applicable to an all-words task. The results show that overall, the system is able to improve the performance of the topical feature set, using the NB algorithm. This could be useful when the local contexts are not reliable, as could happen with automatically acquired features (cf. chapter VI). Another case where the recall is improved is for adjectives, with a gain of 3% recall.

  All in all, the experiments suggest that other ways should be tried to benefit from these features. Instead of the "bag-of-synsets" approach, the usage of dependency relations seems a better way to explore semantic generalization.

- **Selectional preferences:**

  We tested the performance of two models in object/subject relations: word-to-class, and class-to-class. The goal was to disambiguate the nouns in the relations. The two main experiments, which were performed for a sample of nouns, and for all the nouns in four Semcor files, took us to the following conclusions:

  - The class-to-class model obtains better recall than the word-to-class model, with only a small loss in precision. Class-to-class learns selectional preferences for senses of verbs that do not occur in the corpus, via inheritance.

  - The recall of the class-to-class model gets close to the MFS baseline. We have to note that this is a hard baseline for this kind of all-words systems, as we have seen in our study of the literature (cf. section IV.2.2).

  - The preferences are acquired from a small set of tagged examples, and for some words the results are very low. The words with more examples to train show better performance.

The main limitation of the selectional preference approach was the low coverage, and also that no cut-off values or smoothing is applied, and the algorithm is forced to make decisions with few data. Applying a threshold could help to improve precision at the cost of coverage. There are other experiments we would like to explore at this point: the use of a big untagged corpus to learn the preferences, the disambiguation of words with other PoS than nouns, and the inclusion of other features than object and subject.

Sparse data problem and smoothing (5th chapter):

In this chapter we studied different smoothing techniques and ML methods. We built an ensemble of ML methods that was evaluated on Senseval-2 and Senseval-3, and applied to Basque and English.

- **Evaluation on Senseval-2:** The evaluation on Senseval-2 data indicated that the smoothing method explored in this chapter is able to make all three methods perform at very high precisions, comparable and in some cases superior to the best result attained in the Senseval-2 competition. We also showed that a simple combination of the methods and a fourth system based on SVM attains the best result for the Senseval-2 competition reported so far (although only in its more successful configuration, as the system was not "frozen" using cross-validation).

- **Evaluation on Senseval-3:** We participated with our system in the English and Basque lexical-sample tasks in Senseval-3. We submitted two systems for each task after tuning on cross-validation: the best ensemble, and the best single method. Our systems obtained good results, very close to the winning systems in both tasks. For English, our disambiguation method shows a similar behavior on the Senseval-2 and the Senseval-3 datasets (both in cross-validation and against the testing part). The ensemble performs best in all cases, followed by VSM. The smoothing methods contribute to increase the recall in both cases. The results for Basque are different, in this case the best single system is SVM, and the combination of methods does not improve the results. For Basque, the profit from the smoothing methods is much lower, and some algorithms (like VSM) seem to perform below the expectations.

- **Smoothing techniques:** For further study, it would be interesting to extend this work to X/Y features for Y greater than 1, and try other grouping criteria, e.g. taking into account the class of the word. We would also like to compare our results to other more general smoothing techniques (Good, 1953; Jelinek and Mercer, 1980; Chen, 1996).

Automatic acquisition of examples to alleviate the knowledge acquisition bottleneck (6th chapter):

We have applied the "monosemous relatives" method to construct automatically a web corpus which we have used to train three systems based on DL: one fully supervised (applying examples from Semcor and the web corpus), one minimally supervised (relying on the distribution of senses in Semcor and the web corpus) and another fully unsupervised (using an automatically acquired sense rank and the web corpus). The systems were tested on the Senseval-2 lexical sample test set.

- **Performance:** We have shown that the fully supervised system combining our web corpus with the examples in Semcor improves over the same system trained on Semcor alone. This improvement is specially noticeable in the nouns that have less than 10 examples in Semcor. Regarding the minimally supervised and fully unsupervised systems, we have shown that they perform well better than the other systems of the same category presented in the Senseval-2 lexical-sample competition. The system can be trained for all nouns in WordNet, using the data collected from the web[1].

- **Importance of bias:** Knowing how many examples are to be fed into the ML system is a key issue. We have explored several possibilities, and we have seen that assigning directly the first sense in a ranking obtained from hand-tagged data (or even with automatic means on raw corpora) can be a good approximation for disambiguation. However, the DL algorithm is always able to improve this heuristic training on the automatically acquired examples.

- **Limitations of the system:** One of the limitations of our system is that it relies only on DL as learning method. In order to improve

[1] This corpus is available at `http://ixa2.si.ehu.es/pub/sensecorpus`

performance, more powerful ML methods could be applied, like the ensemble constructed on chapter V. We would also like to tune the algorithm that chooses the monosemous relatives, giving preference, for instance, to multiwords that contain the target word as in (Leacock *et al.*, 1998). The method could as well benefit from sophisticated tools to acquire examples that are now available, like ExRetriever (Fernandez *et al.*, 2004), which could open the way to examples with less noise and better performance.

Genre/topic of examples and portability (7th chapter):

In this section we studied the one sense per collocation hypothesis for fine-grained sense distributions, and across genre and topic variations. Here we summarize our main conclusions:

- **Fine-grained disambiguation:** This chapter shows that the one sense per collocation hypothesis is weaker for fine-grained word sense distinctions (e.g. those in WordNet): from the 99% precision mentioned for 2-way ambiguities in (Yarowsky, 1993) we drop to 70% figures. These figures could perhaps be improved using more available data.

- **Cross-corpora disambiguation:** We also show that one sense per collocation does hold across corpora, but that collocations vary from one corpus to other, following genre and topic variations. This explains the low results when performing word sense disambiguation across corpora. In fact, we demonstrated that when two independent corpora share a related genre/topic, the WSD results are better. This has considerable impact in future work on WSD, as genre and topic are shown to be crucial parameters. A system trained on a specific genre/topic would have difficulties to adapt to new genre/topics. Besides, methods that try to extend automatically the amount of examples for training need also to account for genre and topic variations.

- **Cross-validation performance and hand-tagging errors:** As a side effect, we have shown that the results on cross-validation WSD exercises, which mix training and test data drawn from the same documents, are higher than those from a more realistic setting. We also

discovered several hand-tagging errors, which distorted extracted collocations. We did not evaluate the extent of these errors, but they certainly affected the performance on cross-corpora tagging.

VIII.3 Further work

There are open research lines in this work that can be explored further. We will describe here the main experiments that we would like to perform in the future.

- **Integration of selectional preferences in the supervised setting:** We think that despite their low coverage, selectional preferences would help to improve the overall performance of a supervised system, although it is not straightforward how to integrate them with other feature types. One possibility would be to include the sense chosen by the selectional preference model in the feature set, in a fashion similar to (Stevenson and Wilks, 1999). The generalization of syntactic dependencies using WordNet offers promising results, as has been shown in (Mihalcea and Faruque, 2004)[2].

- **Smoothing for automatic acquisition of examples:** An interesting application of the smoothing techniques is to detect good features, even in the case of low amounts of training data. These features could be used as seeds to obtain new examples automatically, in a fashion similar to the method applied in chapter VI for monosemous relatives. They could also be integrated in a bootstrapping process using DLs, as in (Yarowsky, 1995b). The DL algorithm is well suited for this task, as it relies on a single piece of evidence (feature) to choose the correct sense, and it has been shown to perform significantly better with smoothing.

- **Automatic acquisition of examples for improved all-words WSD:** As we mentioned in chapter VI, an automatically obtained corpus was compiled for all nouns in WordNet. We would like to apply this resource to be tested in an all-words task. We would focus on the improvement for words with low amounts of hand-tagged data available.

[2]The system "SenseLearner" has been described in section II.8.

- **Adaptation to the domain:** As we have seen in chapter VII, the performance drops when we train the WSD system on one domain and apply it to another. In order to work on different types of corpora, one promising way would be to apply the automatic ranking by McCarthy *et al.* (2004) to determine the bias of the senses, and use this information to determine the number of training examples for each sense to learn. For that, we would require a big database of examples, which could be obtained by the method presented in chapter VI. Finally, in order to extend our work on domain, one of the factors that should be analyzed is the separate influence of the genre and topic variations.

- **Improvements of the Basque system:** Our main conclusion for Basque is that the chosen feature set should be revised, as it is not clear how to represent the context in case of agglutinative languages. Using a "cleaner" feature set would also help the smoothing techniques. Another interesting experiment would be to rely on the relations in the Basque WordNet to obtain an all-words sense-tagged corpus automatically.

- **Application of high-precision WSD to other tasks:** Regarding the high-precision systems tested on this dissertation, we would like to extend our approaches (based on DLs) to other ML methods. We think that the integration of different high-precision systems could improve the coverage without loss in precision. More importantly, we would like to apply this kind of systems to tasks that could benefit from a partially tagged corpora, like lexical acquisition.

Bibliography

Abe H. and Li N. *Learning Word Association Norms Using Tree Cut Pair Models*. In *Proceedings of the 13th International Conference on Machine Learning, ICML*, 1996.

Abney S. *Bootstrapping*. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, ACL'02*. Philadelphia, 2002.

Aduriz I., Agirre E., Aldezabal I., Alegria I., Arregi X., Arriola J., Artola X., Gojenola K., Maritxalar A., Sarasola K. and Urkia M. *A Word-grammar based morphological analyzer for agglutinative languages*. In *Proceedings of the International Conference on Computational Linguistics COLING*. Saarbrucken, Germany, 2000.

Agirre E., Aldabe I., Lersundi M., Martinez D., Pociello E. and Uria L. *The Basque lexical-sample task*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Agirre E. and Edmonds P. *Machine Learning Methods for Word Sense Disambiguation*. Kluwer Academic Publishers, forthcoming.

Agirre E., Garcia E., Lersundi M., Martinez D. and Pociello E. *The Basque task: did systems perform in the upperbound?*. In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems*. Toulouse, France, 2001.

Agirre E. and Lopez de Lacalle O. *Publicly available topic signatures for all WordNet nominal senses*. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*. Lisbon, Portugal, 2004.

Agirre E. and Martinez D. *Exploring automatic word sense disambiguation with Decision Lists and the Web*. In *Procedings of the COLING 2000 Workshop on Semantic Annotation and Intelligent Content*. Luxembourg, 2000.

Agirre E. and Martinez D. *Decision Lists for English and Basque*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL'2001/EACL'2001*. Toulouse, France, 2001.

Agirre E. and Rigau G. *Word Sense Disambiguation using Conceptual Density*. In *Proceedings of COLING'96*, pp. 16–22. Copenhagen (Denmark), 1996.

Allen J. *Natural Language Understanding*. The Benjamin/Cummings Publishing Company, Inc., 1995.

ALPAC. *Language and Machine: Computers in Translation and Linguistics*. National Research Council Automatic Language Processing Advisory Committee, Washington, DC, 1966.

Argamon-Engelson S. and Dagan I. *Committee–Based Sample Selection for Probabilistic Classifiers*. In *Journal of Artificial Intelligence Research*, volume 11, pp. 335–360, 1999.

Atkins S. *Tools for Computer-Aided Corpus Lexicography: The Hector Project*, pp. 5–72. 41, 1993.

Atserias J., Villarejo L., Rigau G., Agirre E., Carroll J., Magnini B. and Vossen P. *The MEANING Multilingual Central Repository*. In *Proceedings of the 2nd Global WordNet Conference*. Brno, Czech Republic, 2004.

Bar-Hillel Y. *Automatic translation of languages*. Donald Booth and R. E. Meagher, eds., Academic Press, New York, 1960.

Blum A. and Mitchell T. *Combining Labeled and Unlabeled Data with Co-training*. In *Proceedings of the 11h Annual Conference on Computational Learning Theory*, pp. 92–100. ACM Press, New York, 1998.

Brants T. *Tnt - A Statistical Part-of-Speech Tagger*. In *Proceedings of the Sixth Applied Natural Language Processing Conference*. Seattle, WA, 2000.

Carroll J. and Briscoe E. *High precision extraction of grammatical relations*. In *Proceedings of the 7th ACL/SIGPARSE International Workshop on Parsing Technologies*, pp. 78–89. Beijing, China, 2001.

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

Chen S.F. *Building Probabilistic Models for Natural Language*. In *Ph.D. Thesis, Harvard University*, 1996.

Church K. and Gale W. *A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams*. In *Computer Speech and Language*, 5, pp. 19–54, 1991.

Cohen W. *Fast effective rule induction*. In *Proceedings of the 12th International Conference on Machine Learning*. Morgan Kaufmann, 1995.

Collins M. and Singer Y. *Unsupervised Models for Named Entity Classification*. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP/VLC'99*. College Park, MD, USA, 1999.

Crestan E., El-Beze M. and de Loupy C. *Improving WSD with Multi-Level View of Context Monitored by Similarity Measure*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL*. Toulouse, France, 2001.

Cucerzan S. and Yarowsky D. *Minimally Supervised Induction of Grammatical Gender*. In *Proceedings of HLT/NAACL 2003*. Edmonton, 2003.

Daelemans W., Zavrel J., van der Sloot K. and van den Bosch A. *Timbl: Tilburg memory-based learner, version 4.3*. Technical report, Tilburg University and University of Antwerp, 2002.

Dagan I. and Itai A. *Word Sense Disambiguation Using a Second Language Monolingual Corpus*. In *Computational Linguistics 20:4, pp. 563–596.*, 1994.

Dale R., Moisl H. and Somers H. *Handbook of Natural Language Processing*. Marcel Dekker Inc., 2000.

Daude J., Padro L. and Rigau G. *Mapping WordNets Using Structural Information*. In *38th Anual Meeting of the Association for Computational Linguistics (ACL'2000)*. Hong Kong, 2000.

Decadt B., Hoste V. and Daelemans W. *GAMBL, Genetic Algorithm Optimization of Memory-Based WSD*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Dietterich T. *Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms*, pp. 1895–1924. 10 (7), 1998.

Edmonds P. and Cotton S. *SENSEVAL-2: Overview*. In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems*. Toulouse, France, 2001.

Edmonds P. and Kilgarriff A. *Natural Language Engineering, Special Issue on Word Sense Disambiguation Systems*. 8 (4). Cambridge University Press, 2002.

Escudero G., Màrquez L. and Rigau G. *Boosting Applied to Word Sense Disambiguation*. In *Proceedings of the 12th European Conference on Machine Learning, ECML 2000*. Barcelona, Spain, 2000a.

Escudero G., Màrquez L. and Rigau G. *Naive Bayes and Exemplar-based Approaches to Word Sense Disambiguation Revisited*. In *ECAI 2000*, pp. 421–425, 2000b.

Escudero G., Màrquez L. and Rigau G. *On the Portability and Tuning of Supervised Word Sense Disambiguation Systems*. In *In Proceedings of the joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP/VLC*. Hong Kong, China, 2000c.

Fellbaum C. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

Fernandez J., Castillo M., Rigau G., Atserias J. and Turmo J. *Automatic Acquisition of Sense Examples using ExRetriever*. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC)*. Lisbon, Portugal, 2004.

Fernandez-Amoros D., Gonzalo J. and Verdejo F. *The UNED Systems at Senseval-2*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL*. Toulouse, France, 2001.

Firth J.R. *Modes of meaning (1951)*. In J.R. Firth, ed., *Papers in linguistics 1934-1951*, pp. 190–215. Oxford University Press, Oxford, 1957.

Florian R., Cucerzan S., Schafer C. and Yarowsky D. *Combining Classifiers for Word Sense Disambiguation*. In *Journal of Natural Language Engineering*, 8 (4). Cambridge University Press, 2002.

Francis W. and Kucera H. *Brown Corpus Manual of Information*. Department of Linguistics, Brown University, 1964.

Francis W. and Kucera H. *Frequency Analysis of English Usage: Lexicon and Grammar*. Houghton Mifflin, 1982.

Freund Y. and Schapire R. *A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting*, pp. 119–139. 55(1), 1997.

Fujii A., Inui K., Tokunaga T. and Tanaka H. *Selective Sampling for Example-based Word Sense Disambiguation*. In *Computational Linguistics*, 24(4), pp. 573–598, 1998.

Gale W., Church K. and Yarowsky D. *A Method for Disambiguating Word Senses in a Large Corpus*, pp. 415–439. 26, 1993.

Gliozzo A., Magnini B. and Strapparava C. *Unsupervised Domain Relevance Estimation for Word Sense Disambiguation*. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain, 2004.

Good I.J. *The Population Frequencies of Species and the Estimation of Population Parameters*. Biometrika, volume 40, pp. 237–264, 1953.

Grozea C. *Finding optimal parameters for high performance word sense disambiguation*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Hawkins P. and Nettleton D. *Large Scale WSD Using Learning Applied to SENSEVAL*, pp. 135–140. 34 (2), 2000.

Hoste V., Hendrickx I., Daelemans W. and van den Bosch A. *Parameter Optimization for Machine-Learning of Word Sense Disambiguation*. In *Natural Language Engineering, Special Issue on Word Sense Disambiguation Systems*, 8 (4), pp. 311–325, 2002.

Hoste V., Kool A. and Daelemans W. *Classifier Optimization and Combination in the English All Words Task*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL*. Toulouse, France, 2001.

Ide N. and Veronis J. *Introduction to the Special Issue on Word Sense Disambiguation: The State of the Art*. Computational Linguistics, volume 24(1), pp. 1–40, 1998.

Jelinek F. and Mercer R. *Interpolated estimation of Markov source parameters from sparse data*, pp. 381–397. Amsterdam : North Holland Publishing Co., 1980.

Joachims T. *Making Large–Scale SVM Learning Practical*. In B. Schölkopf, C.J.C. Burges and A.J. Smola, eds., *Advances in Kernel Methods — Support Vector Learning*, pp. 169–184. MIT Press, Cambridge, MA, 1999.

Jurafsky D. and Martin J. *An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice-Hall, Upper Saddle River, NJ 07458, 2000.

Karlgren J. and Cutting D. *Recognizing Text Genres with Simple Metrics Using Discriminant Analysis*. In *Proceedings of the International Conference on Computational Linguistics COLING*. Kyoto, 1994.

Katz S. *Estimation of probabilities from sparse data for the language model component of a speech recognizer*. In *IEEE Transactions on Acoustics Speech and Signal Processing*, 1987.

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

Kilgarriff A. *SENSEVAL: An exercise in evaluating word sense disambiguation programs*. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, pp. 581–588. Granada, Spain, 1998.

Kilgarriff A. *English Lexical Sample Task Description*. In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems*. Toulouse, France, 2001.

Kilgarriff A. and Rosenzweig J. *Framework and Results for English SENSEVAL*, pp. 15–48. 34 (2), 2000.

Kilgarriff A. and Tugwell D. *Sketching words*. In *Lexicography and Natural Language Processing: A Festschrift in Honour of B.T.S. Atkins*, 2002.

Krovetz R. *More Than One Sense Per Discourse*. In *Proceedings of SENSEVAL and the Lexicography Loop Workshop*, 1998.

Kunze C. and Lemnitzer L. *Standardizing Wordnets in a Web-compliant Format: The Case of GermaNet*. In *Proceedings of LREC 2002 Workshop on Wordnet Structures and Standardisation, and how these affect Wordnet Applications and Evaluations*, pp. 24–29. Las Palmas de Gran Canaria, Spain, 2002.

Leacock C., Chodorow M. and Miller G.A. *Using Corpus Statistics and WordNet Relations for Sense Identification*. In *Computational Linguistics*, volume 24, pp. 147–165, 1998.

Lee Y. and Ng H. *An Empirical Evaluation of Knowledge Sources and Learning Algorithms for Word Sense Disambiguation*. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pp. 41–48, 2002.

Lesk M. *Automated sense disambiguation using machine-readable dictionaries: How to tell a pinecone from an ice cream cone*. In *Proceedings of the SIGDOC Conference*, 1986.

Lin D. *Automatic retrieval and clustering of similar words*. In *In Proceedings of COLING-ACL*. Montreal, Canada, 1998a.

Lin D. *Dependency-based Evaluation of MINIPAR*. In *Workshop on the Evaluation of Parsing Systems*. Granada, Spain, 1998b.

Magnini B. and Cavagliá G. *Integrating subject field codes into WordNet*. In *Proceedings of the Second International LREC Conference*. Athens, Greece, 2000.

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

Martinez D. and Agirre E. *One Sense per Collocation and Genre/Topic Variations*. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*. Hong Kong, 2000.

McCarthy D., Carroll J. and Preiss J. *Disambiguating Noun and Verb Senses Using Automatically Acquired Selectional Preferences*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL'2001/EACL'2001*. Toulouse, France, 2001.

McCarthy D., Koeling R., Weeds J. and Carroll J. *Finding Predominant Word Senses in Untagged Text*. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*. Barcelona, Spain, 2004.

Mihalcea R. *Bootstrapping Large Sense Tagged Corpora*. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation (LREC)*. Las Palmas, Spain, 2002.

Mihalcea R. and Chklovski T. *Open Mind Word Expert: Creating Large Annotated Data Collections with Web Users' Help*. In *Proceedings of the EACL 2003 Workshop on Linguistically Annotated Corpora (LINC 2003)*. Budapest, Hungary, 2003.

Mihalcea R., Chklovski T. and Killgariff A. *The Senseval-3 English lexical sample task*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Mihalcea R. and Edmonds P. *Senseval-3, Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*. The Association for Computational Linguistics, 2004.

Mihalcea R. and Faruque E. *SenseLearner: Minimally Supervised Word Sense Disambiguation for All Words in Open Text*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Mihalcea R. and Moldovan D. *An Automatic Method for Generating Sense Tagged Corpora*. In *Proceedings of AAAI-99*. Orlando, FL, 1999.

Mihalcea R. and Moldovan D. *Pattern Learning and Active Feature Selection for Word Sense Disambiguation*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL*. Toulouse, France, 2001.

Miller G., Chodorow M., Landes S., Leacock C. and Thomas R. *Using a semantic concordance for sense identification*. In *Proceedings of the ARPA Human Language Technology Workshop*. Morgan Kaufman, San Francisco, 1994.

Miller G.A., Leacock C., Tengi R. and Bunker R. *A Semantic Concordance*. In *Proceedings of the ARPA Human Language Technology Workshop*, pp. 303–308. Princeton, NJ, 1993. Distributed as *Human Language Technology* by San Mateo, CA: Morgan Kaufmann Publishers.

Mooney R. *Comparative Experiments on Disambiguating Word Senses: An Illustration of the Role of Bias in Machine Learning*. In *Proceedings of the 1st Conference on Empirical Methods in Natural Language Processing, EMNLP'96*, pp. 82–91, 1996.

Neter J., Wasserman W. and Kutner M.H. *Applied Linear Statistical Models*. Irwin, Homewood, Illinois, 1985.

Ng H. and Lee Y. *Integrating Multiple Knowledge Sources to Disambiguate Word Sense: An Exemplarbased Approach*. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, 1996.

Ng H., Lim C. and Foo S. *A Case Study on Inter-Annotator Agreement for Word Sense Disambiguation*. In *Proceedings of the ACL SIGLEX Workshop on Standardizing Lexical Resources (SIGLEX99)*, pp. 9–13. College Park, Maryland, USA, 1999.

Ng H.T. *Exemplar-Based Word Sense Disambiguation: Some Recent Improvements*. In C. Cardie and R. Weischedel, eds., *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pp. 208–213. Association for Computational Linguistics, Somerset, New Jersey, 1997.

Ngai G. and Florian R. *Transformation-Based Learning in the Fast Lane*. Proceedings of the Second Conference of the North American Chapter of the Association for Computational Linguistics, pages 40-47, Pittsburgh, PA, USA, 2001.

Palmer M., Fellbaum C., Cotton S., Delfs L. and Dang H. *English Tasks: All-words and Verb Lexical Sample*. In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems*. Toulouse, France, 2001.

Pedersen T. *A Decision Tree of Bigrams is an Accurate Predictor of Word Sense*. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-01)*. Pittsburgh, PA, 2001.

Ravin Y. and Leacock C. *Polysemy: Theoretical and Computational Approaches*. Oxford University Press, 2001.

Resnik P. *A class-based approach to lexical discovery*. In *Proceedings of the Proceedings of the 30th Annual Meeting of the Association for Computational Linguists*, pp. 327–329, 1992.

Resnik P. *Selectional Preference and Sense Disambiguation*. In *Proceedings of the ANLP Workshop "Tagging Text with Lexical Semantics: Why What and How?"*. Washington, DC., 1997.

Resnik P. and Yarowsky D. *A perspective on word sense disambiguation methods and their evaluation*. In *ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How? in conjunction with ANLP-97*. Washington, D.C., USA, 1997.

Sarasola I. *Euskal Hiztegia*. Gipuzkoako Kutxa, Donostia, 1996.

Schapire R. and Singer Y. *Improved Boosting Algorithms Using Confidence-rated Predictions*, pp. 297–336. 37(3), 1999.

Sleator D. and Temperley D. *Parsing English with A Link Grammar*. In *Third Annual Workshop on Parsing Technologies*, 1993.

Snyder B. and Palmer M. *The English all-words task*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Stetina J., Kurohashi S. and Nagao M. *General Word Sense Disambiguation Method Based on a Full Sentential Context*. In *Usage of WordNet in Natural Language Processing , Proceedings of COLING-ACL Workshop*. Montreal (C Canada), 1998.

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

Stevenson M. and Wilks Y. *Combining Weak Knowledge Sources for Sense Disambiguation*. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-99)*. Stockholm, Sweden, 1999.

Stevenson M. and Wilks Y. *The Interaction of Knowledge Sources in Word Sense Disambiguation*. In *Computational Linguistics*, 27(3), 2001.

Strapparava C., Gliozzo A. and Giuliano C. *Pattern Abstraction and Term Similarity for Word Sense Disambiguation: Irst at Senseval-3*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Sussna M. *Word Sense Disambiguation for Free-text Indexing Using a Massive Semantic Network*. In *Proceedings of the Second International Conference on Information and knowledge Management*. Arlington, Virginia USA, 1993.

Tugwell D. and Kilgarriff A. *Wasp-bench: a Lexicographic Tool Supporting Word Sense Disambiguation*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL-2001/EACL-2001*. Toulouse, France, 2001.

Vapnik V.N. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

Veenstra J., van den Bosch A., Buchholz S., Daelemans W. and Zavrel A. *Memory-Based Word Sense Disambiguation*, pp. 171–177. 34 (2), 2000.

Villarejo L., Màrquez L., Agirre E., Martinez D., Magnini B., Strapparava C., McCarthy D., Montoyo A. and Suarez A. *The Meaning System on the English Allwords Task*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain, 2004.

Vossen P. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers, Dordrecht, 1998.

Yarowsky D. *Word-sense disambiguation using statistical models of Roget's categories trained on large corpora*. In *Proceedings of COLING'92*. Nantes, France, 1992.

Yarowsky D. *One Sense per Collocation*. In *Proceedings of the 5th DARPA Speech and Natural Language Workshop*, 1993.

Yarowsky D. *Decision Lists for Lexical Ambiguity Resolution: Application to Accent Restoration in Spanish and French*. In *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, pp. 88–95. Las Cruces, NM, 1994.

Yarowsky D. *Three Machine Learning Algorithms for Lexical Ambiguity Resolution*. In *PhD thesis, Department of Computer and Information Sciences, University of Pennsylvania*, 1995a.

Yarowsky D. *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods*. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189–196. Cambridge, MA, 1995b.

Yarowsky D. *Hierarchical Decision Lists for Word Sense Disambiguation*, pp. 179–186. 34 (2), 2000.

Yarowsky D., Cucerzan S., Florian R., Schafer C. and Wicentowski R. *The Johns Hopkins SENSEVAL2 System Descriptions*. In *Proceedings of SENSEVAL-2*. Toulouse, 2001.

Yarowsky D. and Florian R. *Evaluating Sense Disambiguation Performance Across Diverse Parameter Spaces*. In *Journal of Natural Language Engineering*, 8 (4), pp. 293–310. Cambridge University Press, 2002.

Yngve V. *Syntax and the problem of multiple meaning*. W. Locke and D. Booth, eds., Wiley, New York, 1955.

A. APPENDIX

Publications stemming from this PhD thesis

- Chapter 3:
  - Agirre E. and Martinez D. *Exploring Automatic Word Sense Disambiguation with Decision Lists and the Web*. In *Procedings of the COLING 2000 Workshop on Semantic Annotation and Intelligent Content*. Luxembourg. 2000.
  - Agirre E. and Martinez D. *Decision Lists for English and Basque*. In *Proceedings of the SENSEVAL-2 Workshop. In conjunction with ACL'2001/EACL'2001*. Toulouse, France. 2001.
- Chapter 4:
  - Agirre E. and Martinez D. *Learning class-to-class selectional preferences*. In *Proceedings of the Workshop "Computational Natural Language Learning" (CoNLL-2001). In conjunction with ACL'2001/EACL'2001*. Toulouse, France. 2001.
  - Agirre E. and Martinez D. *Knowledge sources for Word Sense Disambiguation*. In *Proceedings of the Fourth International Conference TSD 2001*. Plzen (Pilsen), Czech Republic. *Published in the Springer Verlag Lecture Notes in Computer Science series. Václav Matousek, Pavel Mautner, Roman Moucek, Karel Tauser (eds.) Copyright Springer-Verlag*. 2001.

  - Agirre E. and Martinez D. *Integrating Selectional Preferences in WordNet*. In *Proceedings of First International WordNet Conference*. Mysore (India). 2002.

  - Martinez D., Agirre.E and Marquez L. *Syntactic Features for High Precision Word Sense Disambiguation*. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING 2002)*. Taipei, Taiwan. 2002.

- Chapter 5:

  - Agirre E. and Martinez D. *Smoothing and Word Sense Disambiguation*. In *Proceedings of EsTAL - España for Natural Language Processing*. Alicante, Spain. 2004.

  - Agirre E. and Martinez D. *The Basque Country University system: English and Basque tasks*. In *Proceedings of the 3rd ACL workshop on the Evaluation of Systems for the Semantic Analysis of Text (SENSEVAL)*. Barcelona, Spain. 2004.

- Chapter 6:

  - Agirre E. and Martinez D. *Unsupervised WSD based on automatically retrieved examples: The importance of bias*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Barcelona, Spain. 2004.

  - Agirre E. and Martinez D. *The effect of bias on an automatically-built word sense corpus*. In *Proceedings of the 4th International Conference on Language Resources and Evaluations (LREC)*. Lisbon, Portugal. 2004.

- Chapter 7:

  - Martinez D. and Agirre E. *One sense per collocation and genre/topic variations*. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*. Hong Kong, 2000.

B. APPENDIX

Additional tables

B.1 Word Sets

Nouns		Verbs		Adjectives		Indeterminates	
-n	N	-v	N	-a	N	-p	N
accident	267	amaze	70	brilliant	229	band	302
behaviour	279	bet	177	deaf	122	bitter	373
bet	274	bother	209	floating	47	hurdle	323
disability	160	bury	201	generous	227	sanction	431
excess	186	calculate	217	giant	97	shake	356
float	75	consume	186	modest	270		
giant	118	derive	216	slight	218		
knee	251	float	229	wooden	195		
onion	214	invade	207				
promise	113	promise	224				
rabbit	221	sack	178				
sack	82	scrap	186				
scrap	156	seize	259				
shirt	184						
steering	176						
TOTAL	2756	TOTAL	2501	TOTAL	1406	TOTAL	1785

Table B.1: The 41 words selected for the English task in Senseval-1, their distribution according to PoS, and the numbers of test instances associated with each (N). Source: *http://www.senseval.org*.

Word	PoS	Senses	Examples	Word	PoS	Senses	Examples
blind	A	7	55	nature	N	7	46
colourless	A	2	35	post	N	13	79
cool	A	6	52	restraint	N	8	45
faithful	A	3	23	sense	N	9	53
fine	A	11	70	spade	N	6	33
fit	A	3	29	stress	N	6	39
free	A	17	82	yew	N	3	28
graceful	A	2	29	begin	V	8	280
green	A	17	94	call	V	23	66
local	A	2	38	carry	V	27	66
natural	A	23	103	collaborate	V	2	30
oblique	A	3	29	develop	V	15	69
simple	A	6	66	draw	V	32	41
solemn	A	2	25	dress	V	14	59
vital	A	7	38	drift	V	9	32
art	N	17	98	drive	V	15	42
authority	N	9	92	face	V	7	93
bar	N	20	151	ferret	V	1	1
bum	N	4	45	find	V	17	68
chair	N	7	69	keep	V	27	67
channel	N	8	73	leave	V	14	66
child	N	7	64	live	V	10	67
church	N	6	64	match	V	8	42
circuit	N	14	85	play	V	25	66
day	N	16	145	pull	V	33	60
detention	N	4	32	replace	V	4	45
dyke	N	2	28	see	V	21	69
facility	N	5	58	serve	V	12	51
fatigue	N	6	43	strike	V	26	54
feeling	N	5	51	train	V	9	63
grip	N	6	51	treat	V	6	44
hearth	N	3	32	turn	V	43	67
holiday	N	6	31	use	V	7	76
lady	N	8	53	wander	V	4	50
material	N	16	69	wash	V	13	12
mouth	N	10	60	work	V	21	60
nation	N	4	37				

Table B.2: The 73 words selected for the English lexical-sample task in Senseval-2. For each word and PoS, the number of senses and the number of testing examples is given (there is approximately twice as much for training each word). Data available at *http://www.senseval.org*.

Word	PoS	Senses	Examples	Word	PoS	Senses	Examples
apal	A	4	44	koroa	N	4	31
arin	A	8	47	lantegi	N	4	42
astun	A	6	56	masa	N	3	36
automatiko	A	2	39	tentsio	N	2	51
borrokalari	A	2	37	ur	N	3	47
gordin	A	5	43	altxatu	V	7	44
hotz	A	6	36	azaldu	V	2	36
natural	A	3	35	baliatu	V	3	36
xehe	A	6	35	ebaki	V	4	37
zahar	A	5	36	edan	V	4	35
bide	N	13	50	ekarri	V	6	52
egun	N	8	53	erre	V	3	47
eliza	N	7	36	etorri	V	6	47
enplegu	N	3	43	galdu	V	7	57
gai	N	3	55	garbitu	V	5	36
herri	N	14	51	gidatu	V	3	53
ibilbide	N	4	49	ikusi	V	7	51
kanal	N	3	37	iraun	V	4	47
kantu	N	5	47	jaio	V	2	38
kapitain	N	3	42	jantzi	V	3	48

Table B.3: The 40 words selected for the Basque lexical-sample task in Senseval-2. For each word and PoS, the number of senses and the number of testing examples is given (there is approximately twice as much for training). Data available at *http://www.senseval.org*.

Word	PoS	Examples	Word	PoS	Examples
different	A	50	begin	V	79
hot	A	43	climb	V	67
important	A	19	decide	V	62
simple	A	18	eat	V	87
solid	A	29	encounter	V	65
argument	N	111	expect	V	78
arm	N	133	express	V	55
atmosphere	N	81	hear	V	32
audience	N	100	lose	V	36
bank	N	132	mean	V	40
degree	N	128	miss	V	30
difference	N	114	note	V	67
difficulty	N	23	operate	V	18
disc	N	100	play	V	52
image	N	74	produce	V	94
interest	N	93	provide	V	69
judgment	N	32	receive	V	27
organization	N	56	remain	V	70
paper	N	117	rule	V	30
party	N	116	smell	V	55
performance	N	87	suspend	V	64
plan	N	84	talk	V	73
shelter	N	98	treat	V	57
sort	N	96	use	V	14
source	N	32	wash	V	34
activate	V	114	watch	V	51
add	V	132	win	V	39
appear	V	133	write	V	23
ask	V	131			

Table B.4: The 57 words selected for the English lexical-sample task in Senseval-3. For each word and PoS, the number of testing examples is given (there is approximately twice as much for training). Data available at *http://www.senseval.org*.

Word	PoS	Examples	Word	PoS	Examples
apal	A	60	koroa	N	66
arin	A	93	lantegi	N	55
astun	A	70	masa	N	45
automatiko	A	45	tentsio	N	63
borrokalari	A	14	ur	N	47
gordin	A	82	azaldu	V	45
hotz	A	48	baliatu	V	37
natural	A	69	berdindu	V	107
xehe	A	37	edan	V	41
zahar	A	73	entrenatu	V	66
bide	N	62	etorri	V	49
egun	N	64	galdu	V	85
eliza	N	42	gidatu	V	60
enplegu	N	43	igo	V	75
gai	N	66	ikusi	V	92
herri	N	83	irabazi	V	56
ibilbide	N	50	iraun	V	56
kanal	N	68	jaio	V	47
kantu	N	45	jaitsi	V	68
kapitain	N	55	jokatu	V	71

Table B.5: The 40 words selected for the Basque lexical-sample task in Senseval-3. For each word and PoS, the number of testing examples is given (there is approximately twice as much for training). Data available at *http://www.senseval.org*.

set A			set B			set C		
Word	PoS	Sense #	Word	PoS	Sense #	Word	PoS	Sense #
All	A	2	Age	N	5	Age	N	5
Long	A	10	Church	N	3	Art	N	4
Most	B	3	Head	N	30	Body	N	9
Only	B	7	Interest	N	7	Car	N	5
Account	N	10	Member	N	5	Child	N	6
Age	N	5	Fall	V	32	Cost	N	3
Church	N	3	Give	V	45	Head	N	28
Duty	N	3	Know	V	11	Interest	N	8
Head	N	30				Line	N	28
Interest	N	7				Point	N	20
Member	N	5				State	N	6
People	N	4				Thing	N	11
Die	V	11				Work	N	6
Fall	V	32				Become	V	4
Give	V	45				Fall	V	17
Include	V	4				Grow	V	8
Know	V	11				Lose	V	10
Seek	V	5				Set	V	20
Understand	V	5				Speak	V	5
						Strike	V	17
						Tell	V	8

Table B.6: Sets of words A, B, and C (cf. section III.3.1.1). The different PoS are separated by horizontal lines.

B.2 Semantic files

Nouns	**Verbs**
Tops	body
act	change
animal	cognition
artifact	communication
attribute	competition
body	consumption
cognition	contact
communication	creation
event	emotion
feeling	motion
food	perception
group	possession
location	social
motive	stative
object	weather
person	
phenomenon	
plant	
possession	
process	
quantity	
relation	
shape	
state	
substance	
time	

Table B.7: List of Semantic Files in WordNet (version 1.7) for nouns and verbs.

B.3 Complete list of relations from Minipar

Table B.8: Complete list of relations from Minipar. For each relation we indicate his type, give a short description, and some examples and comments. We distinguish four kinds of relations: "Rel" (main relation, the relations that seem more useful for disambiguation), "Aux" (auxiliar relations: auxiliar verbs, clauses, etc.), "Fun" (relations that seem irrelevant, but could help on disambiguation), "No" (relations that seem useless for disambiguation).

Relation	Rel	Aux	Fun	No	Description	Examples	Comments
abbrev			x		Abbreviation	NMR → Nuclear ...	
age			x		Age	John, 7, ...	
amod	x				Adverbial modif.	Well thought Merely provide	
appo			x		Apposition	John, director general, ...	
appo-mod			x		Apposition modif.		Often wrong
as-arg				x			
as1				x			
as2				x			
aux		x			Aux. Verb	John should be promoted John -s→ resign ←aux- should ←be- be	
be		x			"be" as aux. Verb	is ←be- sleeping	
being		x			"being" as aux verb		
by-subj	x				Subj. with passives		
c		x			Clausal complement	... that ←c- John loves Mary I go there for + infinitive clause go ←mod- (inf) ←c- for ←i- mainverb	
cn		x?			Nominalized clause	to issue is great be ←s inf ←cn inf ←i issue	Often wrong
comp1	x				Complement (PP, inf/fin clause) of noun	... one of the boys one (N_P) ←comp1- of ← pcomp-n- boy .. grants to finance hospitals grants (N_C) ← c1- (inf) ←i- finance	"boy in the garage" is MOD

Continued on next page

Table B.8: Complete list of relations from Minipar. For each relation we indicate his type, give a short description, and some examples and comments. We distinguish four kinds of relations: "Rel" (main relation, the relations that seem more useful for disambiguation), "Aux" (auxiliar relations: auxiliar verbs, clauses, etc.), "Fun" (relations that seem irrelevant, but could help on disambiguation), "No" (relations that seem useless for disambiguation).

Relation	Rel	Aux	Fun	No	Description	Examples	Comments
						... resolution which voted ... resolution (N_C) ←c1- (fin) ←i- voted	
comp2	x				?????		Few occurrences
conj		x			Conjunction		Indirectly, to find obj
desc	x				Description	... make a man a child make ←desc- child	Occurs frequently
						.... become eclectic	
dest			x		Destination		Often wrong
det	x				Determinant		
expletive	x				It, ...	it was disclosed it -exp→ disclose	
						it means, it seems	
fc	x				Finite complement(?)	... said there is ... say ←fc- (fin) ←i- mainverb	
gen	x				Genitive	court's -gen→ ward	
guest	x				Adjunts(?)	make house ←g- at church	
have		x			"have" as aux. Verb		
head		x			Dep. between query and main verb	should I go.... Q ←inv-aux- should ← head-go	
i		x			See c and fc, dep. between clause and main verb		
inside			x				
inv-aux		x			See head		
inv-be		x					
inv-have		x					
lex-dep	?	?			??	rep., mayor, Mr. ...	It has errors

Continued on next page

Table B.8: Complete list of relations from Minipar. For each relation we indicate his type, give a short description, and some examples and comments. We distinguish four kinds of relations: "Rel" (main relation, the relations that seem more useful for disambiguation), "Aux" (auxiliar relations: auxiliar verbs, clauses, etc.), "Fun" (relations that seem irrelevant, but could help on disambiguation), "No" (relations that seem useless for disambiguation).

Relation	Rel	Aux	Fun	No	Description	Examples	Comments
lex-mod	?	?			?? Multi-word terms ?	oil-filed filed ←lex-mod- oil	makes a single lexical entry: oil-filed, "edge up", "grand jury"
						to edge up edge← up	
						grand jury jury ←lex-mod grand	
						child welfare service	
						"The Constitution"	
						now and then	
location			x				
mod	x				Modifier	Strikes increase as workers demand... increase ←mod as ←comp1 fin ←i demand	
						raises to cope with situation raise ←mod inf ←i cope ←mod with ←pcomp-n situation	
						lost ←mod- already	
						satisfactory -mod→ condition	
neg				x			
nn	x				noun-noun modifier, see also lex-mod	field services sector secotr ←nn field ←nn service	
obj	x				Object		
obj2	x				Indirect object		Sometimes wrong
p-spec		x			pp specifier	back -p-spec→ to	
pcomp-c	x				Clause of pp	in voting itself in ←pcomp-c vpsc ←i- votig	
pcomp-n	x				Nominal head of pp	in the house in ←pcomp-n house	
pnmod	x				Postnominal mod.	person ←pnmod missing	

Continued on next page

Table B.8: Complete list of relations from Minipar. For each relation we indicate his type, give a short description, and some examples and comments. We distinguish four kinds of relations: "Rel" (main relation, the relations that seem more useful for disambiguation), "Aux" (auxiliar relations: auxiliar verbs, clauses, etc.), "Fun" (relations that seem irrelevant, but could help on disambiguation), "No" (relations that seem useless for disambiguation).

Relation	Rel	Aux	Fun	No	Description	Examples	Comments
poss				x	Only for 's		use gen
post	x				The thing after det	few ideas, the first man	
pre	x				The thig before det	all the men, such men	
pred	x				Predicative (can be A or N)	John is beatuful (fin) ←i- is ←pred beautiful ←subj John	
rel		x			Relative clause	earnings which grow earning ←rel fin ←whn which ←i grow	
s	x				Surface subject, better to use subj		
sc	x				Sentential complement	force John to do force ←sc-do	
self	x				Himself...		
spellout			x				
subj	x						
vrel	x				Passive verb modifier of nouns	fund ←vrel-granted	When "pn-mod", is tagged as adj. (often wrongly), here is tagged as verb
wha				x			
whn				x			
whp				x			

B.4 Performance of single features using Semcor

In the following 4 tables, the results obtained using single features are shown. The first two tables illustrate the results for nouns sorted by precision and recall, respectively. The last two tables are devoted to nouns. Many syntactic features do not appear in the training corpus, and are not included in the tables. Some semantic features that have been tested in other works have not been removed from the tables and appear as "basic" features (win_syn_4w, win_anc_0s, win_anc3_0s, win_hyper_0s, win_level4_0s...).

Table B.9: Results in Semcor for the whole set of features disambiguating nouns (sorted by precision).

Feature	Type	Precision	Coverage	Recall
Mod_Prep_pcomp-n_N_word	indirect	100	3.3	3.3
Mod_Prep_pcomp-n_N_synset	indirect	100	3.1	3.1
Mod_lem	IGR-direct	100	1	1
Mod_synset	IGR-direct	100	1	1
Mod_word	IGR-direct	100	1	1
postI_lem	IGR-direct	100	0.7	0.7
postI_word	IGR-direct	100	0.7	0.7
Has_relat_mod_C_i_VI	indirect	100	0.7	0.7
sI_lem	IGR-direct	100	0.6	0.6
sI_synset	IGR-direct	100	0.6	0.6
sI_word	IGR-direct	100	0.6	0.6
subjI_lem	IGR-direct	100	0.6	0.6
subjI_synset	IGR-direct	100	0.6	0.6
subjI_word	IGR-direct	100	0.6	0.6
has_relat_mod_perI	GR-bigr-direct	100	0.6	0.6
postI_synset	IGR-direct	100	0.5	0.5
has_relat_guestI	GR-bigr-direct	100	0.5	0.5
has_relat_mod_fromI	GR-bigr-direct	100	0.5	0.5
genI_synset	IGR-direct	100	0.4	0.4
objI_lem	IGR-direct	100	0.4	0.4
objI_word	IGR-direct	100	0.4	0.4
has_relat_vrelI	GR-bigr-direct	100	0.4	0.4
has_relat_mod_forI	GR-bigr-direct	100	0.3	0.3
conjI_lem	IGR-direct	100	0.2	0.2
conjI_synset	IGR-direct	100	0.2	0.2
guestI_lem	IGR-direct	100	0.2	0.2
guestI_word	IGR-direct	100	0.2	0.2
nn_lem	IGR-direct	100	0.2	0.2
nn_synset	IGR-direct	100	0.2	0.2
nn_word	IGR-direct	100	0.2	0.2
possI_lem	IGR-direct	100	0.2	0.2
possI_word	IGR-direct	100	0.2	0.2
vrelI_lem	IGR-direct	100	0.2	0.2
vrelI_synset	IGR-direct	100	0.2	0.2
vrelI_word	IGR-direct	100	0.2	0.2

Continued on next page

Table B.9: Results in Semcor for the whole set of features disambiguating nouns (sorted by precision).

Feature	Type	Precision	Coverage	Recall
has_relat_appo	GR-bigr-direct	100	0.2	0.2
has_relat_gen	GR-bigr-direct	100	0.2	0.2
has_relat_mod_asI	GR-bigr-direct	100	0.2	0.2
has_relat_mod_outI	GR-bigr-direct	100	0.2	0.2
has_relat_possI	GR-bigr-direct	100	0.2	0.2
comp1_C_i_V_lem	indirect	100	0.2	0.2
comp1_C_i_V_synset	indirect	100	0.2	0.2
comp1_C_i_V_word	indirect	100	0.2	0.2
comp1_Prep_pcomp-n_NI_lem	indirect	100	0.2	0.2
comp1_Prep_pcomp-n_NI_synset	indirect	100	0.2	0.2
comp1_Prep_pcomp-n_NI_word	indirect	100	0.2	0.2
has_relat_s_CN_cn_C_i_VI	indirect	100	0.2	0.2
mod_Prep_pcomp-n_NI_synset	indirect	96.3	2.8	2.7
obj_word	IGR-direct	95.9	5.1	4.9
mod_Prep_pcomp-n_N_lem	indirect	94.7	4	3.8
modI_synset	IGR-direct	94.1	7.1	6.7
obj_lem	IGR-direct	93.3	6.1	5.7
modI_lem	IGR-direct	92.6	8.5	7.9
modI_word	IGR-direct	92.6	8.5	7.9
mod_Prep_pcomp-n_NI_word	indirect	91.4	2.4	2.2
mod_Prep_pcomp-n_NI_lem	indirect	90	3.1	2.8
obj_synset	IGR-direct	88.7	5.5	4.9
trig_wf_+1	basic	88.1	23.7	20.9
trig_lem_+1	basic	87.8	24.8	21.8
nnI_synset	IGR-direct	86.8	3.1	2.7
nnI_lem	IGR-direct	85.7	4.4	3.8
detI_synset	IGR-direct	85.6	1.4	1.2
nnI_word	IGR-direct	85.4	4.3	3.7
big_wf_+1	basic	84.9	62.3	52.9
win_syn_4w	basic	84.8	41.3	35
big_lem_+1	basic	84.6	65.2	55.2
genI_lem	IGR-direct	84	12	10.1
genI_word	IGR-direct	83.6	11.8	9.9
win_wf_3w	basic	83.3	59.4	49.5
comp1_Prep_pcomp-n_N_lem	indirect	83.3	0.6	0.5
comp1_Prep_pcomp-n_N_word	indirect	83.3	0.6	0.5
trig_wf_0	basic	83.1	25.3	21
trig_lem_0	basic	82.4	25.6	21.1
win_wf_4w	basic	82.1	71.2	58.5
has_relat_postI	GR-bigr-direct	81.5	1.2	1
has_relat_mod_ofI	GR-bigr-direct	80	10.9	8.7
win_syn_50w	basic	79.9	99.9	79.8
win_syn_1s	basic	79.9	98.5	78.7
win_syn_20w	basic	79.8	97.4	77.7
trig_wf_-1	basic	79.1	16.7	13.2
has_relat_obj	GR-bigr-direct	78.9	21.3	16.8
win_lem_50w	basic	78.8	100	78.8
trig_lem_-1	basic	78.7	17.3	13.6
has_relat_mod_inI	GR-bigr-direct	78.7	2.7	2.1
has_relat_comp1_Prep_pcomp-n_N	indirect	78.4	1.5	1.2

Continued on next page

Table B.9: Results in Semcor for the whole set of features disambiguating nouns (sorted by precision).

Feature	Type	Precision	Coverage	Recall
win_lem_0s	basic	78.3	100	78.3
win_syn_0s	basic	78.2	84	65.7
win_lem_1s	basic	78.1	100	78.1
win_lem_20w	basic	77.8	100	77.8
win_lem_4w	basic	77.4	99.8	77.2
win_anc_0s	basic	77.1	97.2	74.9
has_relat_genI	GR-bigr-direct	77.1	15.7	12.1
has_relat_comp1_ofI	GR-bigr-direct	76.9	1.4	1.1
win_anc3_0s	basic	76.8	97.1	74.6
Pred_synset	IGR-direct	76.7	1.8	1.4
Win_wf_0s	basic	76.6	97.4	74.6
Win_hyper_0s	basic	76.5	93.1	71.2
Win_level4_0s	basic	75.9	91	69.1
has_relat_subj	GR-bigr-direct	75.4	16.2	12.2
pcomp-n_lem	IGR-direct	75.1	30	22.5
pcomp-n_word	IGR-direct	75.1	30	22.5
subj_lem	IGR-direct	75	5	3.8
has_relat_pred	GR-bigr-direct	75	2.9	2.2
has_relat_comp1_Prep_pcomp-n_NI	indirect	75	1.3	1
win_sf_20w	basic	74.5	100	74.5
win_sf_4w	basic	74.2	99.9	74.1
subj_word	IGR-direct	74.2	3.7	2.7
has_relat_objI	GR-bigr-direct	74.2	2.8	2.1
s_word	IGR-direct	74.1	3.7	2.7
big_lem_-1	basic	74	70.4	52.1
big_wf_-1	basic	74	63.3	46.8
subj_synset	IGR-direct	73.8	4.6	3.4
trig_subpos_0	basic	73.5	81.6	60
trig_subpos_-1	basic	73.2	65.6	48
win_sf_0s	basic	73	100	73
win_sf_50w	basic	72.7	100	72.7
pred_lem	IGR-direct	72.3	2.6	1.9
has_relat_sI	GR-bigr-direct	72.1	3.5	2.5
win_sf_1s	basic	71.7	100	71.7
trig_subpos_+1	basic	71.3	82.9	59.1
big_subpos_+1	basic	70.8	97	68.7
has_relat_s	GR-bigr-direct	70.7	15.7	11.1
has_relat_nnI	GR-bigr-direct	70.4	10.2	7.2
pred_Prep_pcomp-n_N_word	indirect	70	0.5	0.4
has_relat_nn	GR-bigr-direct	69.6	2.4	1.7
pred_word	IGR-direct	69.2	1.3	0.9
detI_word	IGR-direct	69.1	27.9	19.3
detI_lem	IGR-direct	68.9	28.5	19.6
s_lem	IGR-direct	68.8	4.7	3.2
has_relat_subjI	GR-bigr-direct	68.8	4.1	2.8
has_relat_mod	GR-bigr-direct	66.7	1.6	1.1
has_relat_mod_withI	GR-bigr-direct	66.7	0.6	0.4
has_relat_comp1_C_i_V	indirect	66.7	0.6	0.4
trig_pos_+1	basic	66.6	96.7	64.4
trig_pos_0	basic	66.3	97.3	64.5

Continued on next page

Table B.9: Results in Semcor for the whole set of features disambiguating nouns (sorted by precision).

Feature	Type	Precision	Coverage	Recall
trig_pos_-1	basic	64.8	82.8	53.7
big_pos_+1	basic	64.2	99.6	63.9
has_relat_mod_Prep_pcomp-n_N	indirect	63.5	26.4	16.8
s_synset	IGR-direct	63.2	3.6	2.3
big_subpos_-1	basic	63	97.5	61.4
has_relat_detI	GR-bigr-direct	61.5	30.4	18.7
has_relat_pnmodI	GR-bigr-direct	60.8	0.5	0.3
has_relat_by-subj_Prep_pcomp-n_N	indirect	60.8	0.5	0.3
has_relat_modI	GR-bigr-direct	59.1	22.3	13.2
has_relat_conj	GR-bigr-direct	58.6	3.4	2
big_pos_-1	basic	58.4	99.3	58
has_relat_mod_Prep_pcomp-n_NI	indirect	56	20.9	11.7
has_relat_conjI	GR-bigr-direct	55.8	3.7	2.1
has_relat_pcomp-n	GR-bigr-direct	52.1	34.4	17.9
has_relat_mod_onI	GR-bigr-direct	50.2	0.4	0.2
has_relat_lex-mod	GR-bigr-direct	49.6	0.8	0.4
has_relat_appoI	GR-bigr-direct	43.2	0.7	0.3
has_relat_mod_toI	GR-bigr-direct	42.9	0.7	0.3
preI_lem	IGR-direct	33.3	0.3	0.1
preI_word	IGR-direct	33.3	0.3	0.1
has_relat_preI	GR-bigr-direct	33.3	0.3	0.1
comp1_Prep_pcomp-n_N_synset	indirect	33.3	0.3	0.1
has_relat_mod_atI	GR-bigr-direct	29.4	0.7	0.2
has_relat_mod_Prep_pcomp-c_C_i_VI	indirect	20.3	0.5	0.1
has_relat_pred_Prep_pcomp-n_N	indirect	16.7	0.6	0.1
pred_Prep_pcomp-n_N_lem	indirect	16.7	0.6	0.1
pred_Prep_pcomp-n_N_synset	indirect	16.7	0.6	0.1

Table B.10: Results in Semcor for the whole set of features disambiguating verbs (sorted by precision).

Feature	Type	Precision	Coverage	Recall
has_relat_descI	GR-bigr-direct	100	0.3	0.3
conj_synset	IGR-direct	100	0.2	0.2
conjI_lem	IGR-direct	100	0.2	0.2
conjI_synset	IGR-direct	100	0.2	0.2
guestI_synset	IGR-direct	100	0.2	0.2
has_relat_mod_atI	GR-bigr-direct	100	0.2	0.2
has_relat_mod_InI	GR-bigr-direct	100	0.2	0.2
mod_C_i_V_synset	indirect	84.6	1	0.8
sc_lem	IGR-direct	83.3	0.5	0.4
sc_word	IGR-direct	83.3	0.5	0.4
sc_synset	IGR-direct	80	0.4	0.3
mod_C_i_V_lem	indirect	75.2	1.6	1.2
modI_synset	IGR-direct	73.4	1.2	0.9
has_relat_mod_C_i_V	indirect	68.7	6.7	4.6

Continued on next page

Table B.10: Results in Semcor for the whole set of features disambiguating verbs (sorted by precision).

Feature	Type	Precision	Coverage	Recall
has_relat_mod_aboutI	GR-bigr-direct	68.4	1.5	1
has_relat_by-subj_byI	GR-bigr-direct	66.7	0.5	0.3
has_relat_by-subj_Prep_pcomp-n_NI	indirect	66.7	0.5	0.3
has_relat_vrel	GR-bigr-direct	63.4	0.9	0.6
fc_C_i_V_synset	indirect	57.5	1.5	0.9
has_relat_amodI	GR-bigr-direct	56.7	11.3	6.4
fc_C_i_VI_word	indirect	56.7	7.1	4
has_relat_fc_C_i_VI	indirect	56.6	16.4	9.3
mod_C_i_V_word	indirect	56.6	1.1	0.6
modI_word	IGR-direct	55.8	1.9	1.1
fc_C_i_V_word	indirect	55.4	1.4	0.8
amodI_synset	IGR-direct	54.8	7.8	4.3
trig_wf_0	basic	54.3	28.7	15.6
amodI_lem	IGR-direct	54	8.3	4.5
amodI_word	IGR-direct	54	8.3	4.5
sI_synset	IGR-direct	53.7	5.5	3
trig_lem_0	basic	53.3	31.5	16.8
mod_Prep_pcomp-n_NI_word	indirect	52.8	1.8	1
trig_lem_-1	basic	52.7	25.2	13.3
has_relat_sc	GR-bigr-direct	52.2	2.4	1.3
conj_lem	IGR-direct	51.8	0.3	0.2
modI_lem	IGR-direct	51.7	2.1	1.1
fc_C_i_VI_lem	indirect	51.6	8.8	4.5
has_relat_comp1_C_i_V	indirect	50.8	6.9	3.5
subjI_lem	IGR-direct	50.7	40.8	20.7
subjI_word	IGR-direct	50.7	36.9	18.7
big_lem_-1	basic	50.2	75.9	38.1
trig_wf_+1	basic	50	27.3	13.7
pred_C_i_V_word	indirect	50	0.2	0.1
win_lem_20w	basic	49.9	100	49.9
win_lem_50w	basic	49.9	100	49.9
trig_wf_-1	basic	49.9	22.6	11.3
win_hyper_0s	basic	49.8	89.6	44.6
win_syn_50w	basic	49.5	100	49.5
sI_lem	IGR-direct	49.2	39	19.2
win_anc_0s	basic	49.1	93.2	45.8
win_anc3_0s	basic	49	92.9	45.5
win_wf_3w	basic	48.9	61.7	30.2
win_syn_1s	basic	48.8	98.8	48.2
big_wf_-1	basic	48.8	62.8	30.6
trig_lem_+1	basic	48.7	30.7	15
win_lem_0s	basic	48.6	100	48.6
win_syn_20w	basic	48.6	97.8	47.5
win_lem_4w	basic	48.5	100	48.5
win_wf_4w	basic	48.5	75.3	36.5
sI_word	IGR-direct	48.5	35.2	17.1
objI_synset	IGR-direct	48.5	7.2	3.5
subjI_synset	IGR-direct	48.3	6.3	3
win_wf_0s	basic	48.1	94.8	45.6

Continued on next page

Table B.10: Results in Semcor for the whole set of features disambiguating verbs (sorted by precision).

Feature	Type	Precision	Coverage	Recall
win_syn_0s	basic	48	81.1	38.9
win_lem_1s	basic	47.7	100	47.7
win_syn_4w	basic	47.6	47.9	22.8
auxI_lem	IGR-direct	47.4	20.8	9.9
auxI_word	IGR-direct	47.4	20.8	9.9
GR-ngram3	GR-ngram	47.2	66.4	31.3
big_lem_+1	basic	46.4	74.1	34.4
obj2I_synset	IGR-direct	46.4	2.2	1
has_relat_auxI	GR-bigr-direct	46.2	23	10.6
win_sf_4w	basic	46.1	100	46.1
obj2I_lem	IGR-direct	45.9	2.9	1.3
win_level4_0s	basic	45.6	85.9	39.2
win_sf_50w	basic	45.4	100	45.4
win_sf_20w	basic	45.3	100	45.3
GR-ngram1	GR-ngram	45.3	99.7	45.2
big_wf_+1	basic	45	64.8	29.2
win_sf_1s	basic	44.9	100	44.9
win_sf_0s	basic	44.8	98.8	44.3
has_relat_subjI	GR-bigr-direct	44.8	62	27.8
trig_subpos_0	basic	44.7	80.8	36.1
big_subpos_+1	basic	44.3	98	43.4
has_relat_sI	GR-bigr-direct	44.3	60.3	26.7
big_subpos_-1	basic	44.2	98.5	43.5
has_relat_mod_Prep_pcomp-c_C_i_V	indirect	43.9	1.2	0.5
has_relat_mod_asI	GR-bigr-direct	43.6	0.5	0.2
big_pos_+1	basic	43.3	99.6	43.1
trig_pos_0	basic	43.1	98	42.2
trig_subpos_-1	basic	42.7	80.8	34.5
big_pos_-1	basic	42.6	99.5	42.4
trig_pos_-1	basic	42.2	92.2	38.9
has_relat_mod_byI	GR-bigr-direct	42.2	0.6	0.3
GR-ngram2	GR-ngram	41.9	92.7	38.8
trig_pos_+1	basic	41.8	97.6	40.8
has_relat_mod_ofI	GR-bigr-direct	41.6	4.6	1.9
beI_lem	IGR-direct	41.4	5.8	2.4
beI_word	IGR-direct	41.4	5.8	2.4
trig_subpos_+1	basic	41.2	82.8	34.1
objI_lem	IGR-direct	40.5	23.4	9.5
comp1_C_i_V_synset	indirect	40.1	0.8	0.3
fc_C_i_V_lem	indirect	39.6	1.9	0.8
obj2I_word	IGR-direct	39.4	2.6	1
has_relat_fc_C_i_V	indirect	39.2	6.2	2.4
fc_C_i_VI_synset	indirect	38.6	5.8	2.2
mod_Prep_pcomp-n_NI_lem	indirect	38.4	2.5	1
objI_word	IGR-direct	38.3	20.3	7.8
haveI_lem	IGR-direct	36.4	4.8	1.7
haveI_word	IGR-direct	36.4	4.8	1.7
has_relat_s_CN_cn_C_i_V	indirect	34.2	1.8	0.6
has_relat_s_CN_cn_C_i_VI	indirect	33.8	0.6	0.2

Continued on next page

Table B.10: Results in Semcor for the whole set of features disambiguating verbs (sorted by precision).

Feature	Type	Precision	Coverage	Recall
has_relat_mod_onI	GR-bigr-direct	33.3	0.5	0.2
has_relat_mod_intoI	GR-bigr-direct	33.3	0.2	0.1
comp1_C_i_V_lem	indirect	32.9	3.3	1.1
has_relat_beI	GR-bigr-direct	30.1	6.2	1.9
comp1_C_i_V_word	indirect	30.1	2.8	0.8
has_relat_modI	GR-bigr-direct	28.6	7.2	2.1
has_relat_mod_Prep_pcomp-n_NI	indirect	27.8	17.3	4.8
has_relat_haveI	GR-bigr-direct	27.4	5.1	1.4
has_relat_conj	GR-bigr-direct	24.3	2	0.5
has_relat_mod_C_i_VI	indirect	21.3	3.4	0.7
has_relat_objI	GR-bigr-direct	20	52.1	10.4
has_relat_mod_inI	GR-bigr-direct	19.2	3.1	0.6
mod_C_i_VI_synset	indirect	16.7	0.5	0.1
has_relat_conjI	GR-bigr-direct	15.3	2.2	0.3
guestI_lem	IGR-direct	14.7	2.6	0.4
guestI_word	IGR-direct	14.7	2.6	0.4
mod_C_i_VI_word	indirect	14.3	0.6	0.1
mod_Prep_pcomp-n_NI_synset	indirect	13.3	1.2	0.2
pred_C_i_V_lem	indirect	12	0.6	0.1
mod_C_i_VI_lem	indirect	11.1	0.7	0.1
has_relat_pred_C_i_V	indirect	9.5	0.7	0.1
has_relat_guestI	GR-bigr-direct	6.2	6.2	0.4
has_relat_mod_forI	GR-bigr-direct	5.6	1	0.1
has_relat_mod_toI	GR-bigr-direct	2.8	2.3	0.1
has_relat_obj2I	GR-bigr-direct	1.4	9.6	0.1

Table B.11: Results in Semcor for the whole set of features disambiguating nouns (sorted by recall).

Feature	Type	Precision	Coverage	Recall
Win_syn_50w	basic	79.9	99.9	79.8
Win_lem_50w	basic	78.8	100	78.8
win_syn_1s	basic	79.9	98.5	78.7
win_lem_0s	basic	78.3	100	78.3
win_lem_1s	basic	78.1	100	78.1
win_lem_20w	basic	77.8	100	77.8
win_syn_20w	basic	79.8	97.4	77.7
win_lem_4w	basic	77.4	99.8	77.2
win_anc_0s	basic	77.1	97.2	74.9
win_anc3_0s	basic	76.8	97.1	74.6
win_wf_0s	basic	76.6	97.4	74.6
win_sf_20w	basic	74.5	100	74.5
win_sf_4w	basic	74.2	99.9	74.1
win_sf_0s	basic	73	100	73
win_sf_50w	basic	72.7	100	72.7

Continued on next page

Table B.11: Results in Semcor for the whole set of features disambiguating nouns (sorted by recall).

Feature	Type	Precision	Coverage	Recall
win_sf_1s	basic	71.7	100	71.7
win_hyper_0s	basic	76.5	93.1	71.2
win_level4_0s	basic	75.9	91	69.1
big_subpos_+1	basic	70.8	97	68.7
win_syn_0s	basic	78.2	84	65.7
trig_pos_0	basic	66.3	97.3	64.5
trig_pos_+1	basic	66.6	96.7	64.4
big_pos_+1	basic	64.2	99.6	63.9
big_subpos_-1	basic	63	97.5	61.4
trig_subpos_0	basic	73.5	81.6	60
trig_subpos_+1	basic	71.3	82.9	59.1
win_wf_4w	basic	82.1	71.2	58.5
big_pos_-1	basic	58.4	99.3	58
big_lem_+1	basic	84.6	65.2	55.2
trig_pos_-1	basic	64.8	82.8	53.7
big_wf_+1	basic	84.9	62.3	52.9
big_lem_-1	basic	74	70.4	52.1
win_wf_3w	basic	83.3	59.4	49.5
trig_subpos_-1	basic	73.2	65.6	48
big_wf_-1	basic	74	63.3	46.8
win_syn_4w	basic	84.8	41.3	35
pcomp-n_lem	IGR-direct	75.1	30	22.5
pcomp-n_word	IGR-direct	75.1	30	22.5
trig_lem_+1	basic	87.8	24.8	21.8
trig_lem_0	basic	82.4	25.6	21.1
trig_wf_0	basic	83.1	25.3	21
trig_wf_+1	basic	88.1	23.7	20.9
detI_lem	IGR-direct	68.9	28.5	19.6
detI_word	IGR-direct	69.1	27.9	19.3
has_relat_detI	GR-bigr-direct	61.5	30.4	18.7
Has_relat_pcomp-n	GR-bigr-direct	52.1	34.4	17.9
has_relat_obj	GR-bigr-direct	78.9	21.3	16.8
has_relat_mod_Prep_pcomp-n_N	indirect	63.5	26.4	16.8
trig_lem_-1	basic	78.7	17.3	13.6
trig_wf_-1	basic	79.1	16.7	13.2
has_relat_modI	GR-bigr-direct	59.1	22.3	13.2
has_relat_subj	GR-bigr-direct	75.4	16.2	12.2
has_relat_genI	GR-bigr-direct	77.1	15.7	12.1
has_relat_mod_Prep_pcomp-n_NI	indirect	56	20.9	11.7
has_relat_s	GR-bigr-direct	70.7	15.7	11.1
genI_lem	IGR-direct	84	12	10.1
genI_word	IGR-direct	83.6	11.8	9.9
has_relat_mod_ofI	GR-bigr-direct	80	10.9	8.7
modI_lem	IGR-direct	92.6	8.5	7.9
modI_word	IGR-direct	92.6	8.5	7.9
has_relat_nnI	GR-bigr-direct	70.4	10.2	7.2
modI_synset	IGR-direct	94.1	7.1	6.7
obj_lem	IGR-direct	93.3	6.1	5.7
obj_word	IGR-direct	95.9	5.1	4.9

Continued on next page

Table B.11: Results in Semcor for the whole set of features disambiguating nouns (sorted by recall).

Feature	Type	Precision	Coverage	Recall
obj_synset	IGR-direct	88.7	5.5	4.9
mod_Prep_pcomp-n_N_lem	indirect	94.7	4	3.8
nnI_lem	IGR-direct	85.7	4.4	3.8
subj_lem	IGR-direct	75	5	3.8
nnI_word	IGR-direct	85.4	4.3	3.7
subj_synset	IGR-direct	73.8	4.6	3.4
Mod_Prep_pcomp-n_N_word	indirect	100	3.3	3.3
s_lem	IGR-direct	68.8	4.7	3.2
Mod_Prep_pcomp-n_N_synset	indirect	100	3.1	3.1
mod_Prep_pcomp-n_NI_lem	indirect	90	3.1	2.8
has_relat_subjI	GR-bigr-direct	68.8	4.1	2.8
mod_Prep_pcomp-n_NI_synset	indirect	96.3	2.8	2.7
nnI_synset	IGR-direct	86.8	3.1	2.7
subj_word	IGR-direct	74.2	3.7	2.7
s_word	IGR-direct	74.1	3.7	2.7
has_relat_sI	GR-bigr-direct	72.1	3.5	2.5
s_synset	IGR-direct	63.2	3.6	2.3
mod_Prep_pcomp-n_NI_word	indirect	91.4	2.4	2.2
has_relat_pred	GR-bigr-direct	75	2.9	2.2
has_relat_mod_inI	GR-bigr-direct	78.7	2.7	2.1
has_relat_objI	GR-bigr-direct	74.2	2.8	2.1
has_relat_conjI	GR-bigr-direct	55.8	3.7	2.1
has_relat_conj	GR-bigr-direct	58.6	3.4	2
pred_lem	IGR-direct	72.3	2.6	1.9
has_relat_nn	GR-bigr-direct	69.6	2.4	1.7
pred_synset	IGR-direct	76.7	1.8	1.4
detI_synset	IGR-direct	85.6	1.4	1.2
has_relat_comp1_Prep_pcomp-n_N	indirect	78.4	1.5	1.2
has_relat_comp1_ofI	GR-bigr-direct	76.9	1.4	1.1
has_relat_mod	GR-bigr-direct	66.7	1.6	1.1
Mod_lem	IGR-direct	100	1	1
Mod_synset	IGR-direct	100	1	1
mod_word	IGR-direct	100	1	1
has_relat_postI	GR-bigr-direct	81.5	1.2	1
has_relat_comp1_Prep_pcomp-n_NI	indirect	75	1.3	1
pred_word	IGR-direct	69.2	1.3	0.9
postI_lem	IGR-direct	100	0.7	0.7
postI_word	IGR-direct	100	0.7	0.7
has_relat_mod_C_i_VI	indirect	100	0.7	0.7
sI_lem	IGR-direct	100	0.6	0.6
sI_synset	IGR-direct	100	0.6	0.6
sI_word	IGR-direct	100	0.6	0.6
subjI_lem	IGR-direct	100	0.6	0.6
subjI_synset	IGR-direct	100	0.6	0.6
subjI_word	IGR-direct	100	0.6	0.6
has_relat_mod_perI	GR-bigr-direct	100	0.6	0.6
postI_synset	IGR-direct	100	0.5	0.5
has_relat_guestI	GR-bigr-direct	100	0.5	0.5
has_relat_mod_fromI	GR-bigr-direct	100	0.5	0.5

Continued on next page

Table B.11: Results in Semcor for the whole set of features disambiguating nouns (sorted by recall).

Feature	Type	Precision	Coverage	Recall
comp1_Prep_pcomp-n_N_lem	indirect	83.3	0.6	0.5
comp1_Prep_pcomp-n_N_word	indirect	83.3	0.6	0.5
genI_synset	IGR-direct	100	0.4	0.4
objI_lem	IGR-direct	100	0.4	0.4
objI_word	IGR-direct	100	0.4	0.4
has_relat_vrelI	GR-bigr-direct	100	0.4	0.4
pred_Prep_pcomp-n_N_word	indirect	70	0.5	0.4
has_relat_mod_withI	GR-bigr-direct	66.7	0.6	0.4
has_relat_comp1_C_i_V	indirect	66.7	0.6	0.4
has_relat_lex-mod	GR-bigr-direct	49.6	0.8	0.4
has_relat_mod_forI	GR-bigr-direct	100	0.3	0.3
has_relat_pnmodI	GR-bigr-direct	60.8	0.5	0.3
has_relat_by-subj_Prep_pcomp-n_N	indirect	60.8	0.5	0.3
has_relat_appoI	GR-bigr-direct	43.2	0.7	0.3
has_relat_mod_toI	GR-bigr-direct	42.9	0.7	0.3
conjI_lem	IGR-direct	100	0.2	0.2
conjI_synset	IGR-direct	100	0.2	0.2
guestI_lem	IGR-direct	100	0.2	0.2
guestI_word	IGR-direct	100	0.2	0.2
nn_lem	IGR-direct	100	0.2	0.2
nn_synset	IGR-direct	100	0.2	0.2
nn_word	IGR-direct	100	0.2	0.2
possI_lem	IGR-direct	100	0.2	0.2
possI_word	IGR-direct	100	0.2	0.2
vrelI_lem	IGR-direct	100	0.2	0.2
vrelI_synset	IGR-direct	100	0.2	0.2
vrelI_word	IGR-direct	100	0.2	0.2
has_relat_appo	GR-bigr-direct	100	0.2	0.2
has_relat_gen	GR-bigr-direct	100	0.2	0.2
has_relat_mod_asI	GR-bigr-direct	100	0.2	0.2
has_relat_mod_outI	GR-bigr-direct	100	0.2	0.2
has_relat_possI	GR-bigr-direct	100	0.2	0.2
comp1_C_i_V_lem	indirect	100	0.2	0.2
comp1_C_i_V_synset	indirect	100	0.2	0.2
comp1_C_i_V_word	indirect	100	0.2	0.2
comp1_Prep_pcomp-n_NI_lem	indirect	100	0.2	0.2
comp1_Prep_pcomp-n_NI_synset	indirect	100	0.2	0.2
comp1_Prep_pcomp-n_NI_word	indirect	100	0.2	0.2
has_relat_s_CN_cn_C_i_VI	indirect	100	0.2	0.2
has_relat_mod_onI	GR-bigr-direct	50.2	0.4	0.2
has_relat_mod_atI	GR-bigr-direct	29.4	0.7	0.2
preI_lem	IGR-direct	33.3	0.3	0.1
preI_word	IGR-direct	33.3	0.3	0.1
has_relat_preI	GR-bigr-direct	33.3	0.3	0.1
comp1_Prep_pcomp-n_N_synset	indirect	33.3	0.3	0.1
has_relat_mod_Prep_pcomp-c_C_i_VI	indirect	20.3	0.5	0.1
has_relat_pred_Prep_pcomp-n_N	indirect	16.7	0.6	0.1
pred_Prep_pcomp-n_N_lem	indirect	16.7	0.6	0.1
pred_Prep_pcomp-n_N_synset	indirect	16.7	0.6	0.1

Table B.12: Results in Semcor for the whole set of features disambiguating verbs (sorted by recall).

Feature	Type	Precision	Coverage	Recall
win_lem_20w	basic	49.9	100	49.9
win_lem_50w	basic	49.9	100	49.9
win_syn_50w	basic	49.5	100	49.5
win_lem_0s	basic	48.6	100	48.6
win_lem_4w	basic	48.5	100	48.5
win_syn_1s	basic	48.8	98.8	48.2
win_lem_1s	basic	47.7	100	47.7
win_syn_20w	basic	48.6	97.8	47.5
win_sf_4w	basic	46.1	100	46.1
win_anc_0s	basic	49.1	93.2	45.8
win_wf_0s	basic	48.1	94.8	45.6
win_anc3_0s	basic	49	92.9	45.5
win_sf_50w	basic	45.4	100	45.4
win_sf_20w	basic	45.3	100	45.3
GR-ngram1	GR-ngram	45.3	99.7	45.2
win_sf_1s	basic	44.9	100	44.9
win_hyper_0s	basic	49.8	89.6	44.6
win_sf_0s	basic	44.8	98.8	44.3
big_subpos_-1	basic	44.2	98.5	43.5
big_subpos_+1	basic	44.3	98	43.4
big_pos_+1	basic	43.3	99.6	43.1
big_pos_-1	basic	42.6	99.5	42.4
trig_pos_0	basic	43.1	98	42.2
trig_pos_+1	basic	41.8	97.6	40.8
win_level4_0s	basic	45.6	85.9	39.2
win_syn_0s	basic	48	81.1	38.9
trig_pos_-1	basic	42.2	92.2	38.9
GR-ngram2	GR-ngram	41.9	92.7	38.8
big_lem_-1	basic	50.2	75.9	38.1
win_wf_4w	basic	48.5	75.3	36.5
trig_subpos_0	basic	44.7	80.8	36.1
trig_subpos_-1	basic	42.7	80.8	34.5
big_lem_+1	basic	46.4	74.1	34.4
trig_subpos_+1	basic	41.2	82.8	34.1
GR-ngram3	GR-ngram	47.2	66.4	31.3
big_wf_-1	basic	48.8	62.8	30.6
win_wf_3w	basic	48.9	61.7	30.2
big_wf_+1	basic	45	64.8	29.2
has_relat_subjI	GR-bigr-direct	44.8	62	27.8
has_relat_sI	GR-bigr-direct	44.3	60.3	26.7
win_syn_4w	basic	47.6	47.9	22.8
subjI_lem	IGR-direct	50.7	40.8	20.7
sI_lem	IGR-direct	49.2	39	19.2
subjI_word	IGR-direct	50.7	36.9	18.7
sI_word	IGR-direct	48.5	35.2	17.1
trig_lem_0	basic	53.3	31.5	16.8

Continued on next page

Table B.12: Results in Semcor for the whole set of features disambiguating verbs (sorted by recall).

Feature	Type	Precision	Coverage	Recall
trig_wf_0	basic	54.3	28.7	15.6
trig_lem_+1	basic	48.7	30.7	15
trig_wf_+1	basic	50	27.3	13.7
trig_lem_-1	basic	52.7	25.2	13.3
trig_wf_-1	basic	49.9	22.6	11.3
has_relat_auxI	GR-bigr-direct	46.2	23	10.6
has_relat_objI	GR-bigr-direct	20	52.1	10.4
auxI_lem	IGR-direct	47.4	20.8	9.9
auxI_word	IGR-direct	47.4	20.8	9.9
objI_lem	IGR-direct	40.5	23.4	9.5
has_relat_fc_C_i_VI	indirect	56.6	16.4	9.3
objI_word	IGR-direct	38.3	20.3	7.8
has_relat_amodI	GR-bigr-direct	56.7	11.3	6.4
has_relat_mod_Prep_pcomp-n_NI	indirect	27.8	17.3	4.8
has_relat_mod_C_i_V	indirect	68.7	6.7	4.6
amodI_lem	IGR-direct	54	8.3	4.5
amodI_word	IGR-direct	54	8.3	4.5
fc_C_i_VI_lem	indirect	51.6	8.8	4.5
amodI_synset	IGR-direct	54.8	7.8	4.3
fc_C_i_VI_word	indirect	56.7	7.1	4
has_relat_comp1_C_i_V	indirect	50.8	6.9	3.5
objI_synset	IGR-direct	48.5	7.2	3.5
sI_synset	IGR-direct	53.7	5.5	3
subjI_synset	IGR-direct	48.3	6.3	3
beI_lem	IGR-direct	41.4	5.8	2.4
beI_word	IGR-direct	41.4	5.8	2.4
has_relat_fc_C_i_V	indirect	39.2	6.2	2.4
fc_C_i_VI_synset	indirect	38.6	5.8	2.2
has_relat_modI	GR-bigr-direct	28.6	7.2	2.1
has_relat_mod_ofI	GR-bigr-direct	41.6	4.6	1.9
has_relat_beI	GR-bigr-direct	30.1	6.2	1.9
haveI_lem	IGR-direct	36.4	4.8	1.7
haveI_word	IGR-direct	36.4	4.8	1.7
has_relat_haveI	GR-bigr-direct	27.4	5.1	1.4
has_relat_sc	GR-bigr-direct	52.2	2.4	1.3
obj2I_lem	IGR-direct	45.9	2.9	1.3
mod_C_i_V_lem	indirect	75.2	1.6	1.2
modI_word	IGR-direct	55.8	1.9	1.1
modI_lem	IGR-direct	51.7	2.1	1.1
comp1_C_i_V_lem	indirect	32.9	3.3	1.1
has_relat_mod_aboutI	GR-bigr-direct	68.4	1.5	1
mod_Prep_pcomp-n_NI_word	indirect	52.8	1.8	1
obj2I_synset	IGR-direct	46.4	2.2	1
obj2I_word	IGR-direct	39.4	2.6	1
mod_Prep_pcomp-n_NI_lem	indirect	38.4	2.5	1
modI_synset	IGR-direct	73.4	1.2	0.9
fc_C_i_V_synset	indirect	57.5	1.5	0.9
mod_C_i_V_synset	indirect	84.6	1	0.8
fc_C_i_V_word	indirect	55.4	1.4	0.8

Continued on next page

Table B.12: Results in Semcor for the whole set of features disambiguating verbs (sorted by recall).

Feature	Type	Precision	Coverage	Recall
fc_C_i_V_lem	indirect	39.6	1.9	0.8
comp1_C_i_V_word	indirect	30.1	2.8	0.8
has_relat_mod_C_i_VI	indirect	21.3	3.4	0.7
has_relat_vrel	GR-bigr-direct	63.4	0.9	0.6
mod_C_i_V_word	indirect	56.6	1.1	0.6
has_relat_s_CN_cn_C_i_V	indirect	34.2	1.8	0.6
has_relat_mod_inI	GR-bigr-direct	19.2	3.1	0.6
has_relat_mod_Prep_pcomp-c_C_i_V	indirect	43.9	1.2	0.5
has_relat_conj	GR-bigr-direct	24.3	2	0.5
sc_lem	IGR-direct	83.3	0.5	0.4
sc_word	IGR-direct	83.3	0.5	0.4
guestI_lem	IGR-direct	14.7	2.6	0.4
guestI_word	IGR-direct	14.7	2.6	0.4
has_relat_guestI	GR-bigr-direct	6.2	6.2	0.4
has_relat_descI	GR-bigr-direct	100	0.3	0.3
sc_synset	IGR-direct	80	0.4	0.3
has_relat_by-subj_byI	GR-bigr-direct	66.7	0.5	0.3
has_relat_by-subj_Prep_pcomp-n_NI	indirect	66.7	0.5	0.3
has_relat_mod_byI	GR-bigr-direct	42.2	0.6	0.3
comp1_C_i_V_synset	indirect	40.1	0.8	0.3
has_relat_conjI	GR-bigr-direct	15.3	2.2	0.3
conj_synset	IGR-direct	100	0.2	0.2
conjI_lem	IGR-direct	100	0.2	0.2
conjI_synset	IGR-direct	100	0.2	0.2
guestI_synset	IGR-direct	100	0.2	0.2
has_relat_mod_atI	GR-bigr-direct	100	0.2	0.2
has_relat_mod_InI	GR-bigr-direct	100	0.2	0.2
conj_lem	IGR-direct	51.8	0.3	0.2
has_relat_mod_asI	GR-bigr-direct	43.6	0.5	0.2
has_relat_s_CN_cn_C_i_VI	indirect	33.8	0.6	0.2
has_relat_mod_onI	GR-bigr-direct	33.3	0.5	0.2
mod_Prep_pcomp-n_NI_synset	indirect	13.3	1.2	0.2
pred_C_i_V_word	indirect	50	0.2	0.1
has_relat_mod_intoI	GR-bigr-direct	33.3	0.2	0.1
mod_C_i_VI_synset	indirect	16.7	0.5	0.1
mod_C_i_VI_word	indirect	14.3	0.6	0.1
pred_C_i_V_lem	indirect	12	0.6	0.1
mod_C_i_VI_lem	indirect	11.1	0.7	0.1
has_relat_pred_C_i_V	indirect	9.5	0.7	0.1
has_relat_mod_forI	GR-bigr-direct	5.6	1	0.1
has_relat_mod_toI	GR-bigr-direct	2.8	2.3	0.1
has_relat_obj2I	GR-bigr-direct	1.4	9.6	0.1