Euskal Herriko Unibertsitatea

eman ta zabal zazu

Informatika Fakultatea
Lengoaia eta Sistema Informatikoak Saila

Domain-Specific Word Sense Disambiguation

Oier Lopez de Lacalle Lekuonak
Informatikan Doktore titulua eskuratzeko aurkezturiko
Tesi-Txostena

Donostia, 2009ko iraila

Euskal Herriko Unibertsitatea

eman ta zabal zazu

Informatika Fakultatea
Lengoaia eta Sistema Informatikoak Saila

Domain-Specific Word Sense Disambiguation

Oier Lopez de Lacalle Lekuonak Eneko Agirre Bengoaren zuzendaritzapean egindako tesiaren txostena, Euskal Herriko Unibertsitatean Informatikan Doktore titulua eskuratzeko aurkeztua

Donostia, 2009ko iraila

Lan hau Eusko Jaurlaritzaren ikertzaileak prestatzeko beka batekin (BFI05.265) egin dut.

Are you suggesting coconuts migrate?

Monty Python and the Holy Grail

Aita eta amari

Esker onak / Acknowledgements

Inor baino lehen tesi honen zuzendaria den Eneko Agirre eskertu nahi dut. Eneko, eskerrik asko ikerkuntza munduan sartzen laguntzeagtik, eta hau zer den erakusteagatik. Zure energia eta baikortasunagatik ez balitz tesia idazteko nengoke oraindik.

David, zuri ere eskerrak eman beharrean nago. Batez ere tesi hau ez zelako posible izango aurretik ez bazenitu mundu xelebre honetako hainbat ate ireki. Tesi hasierako urteetan asko ikasi nuen zure ondoan.

Tesiko azken kapitulua hein handi batean Aitor Soroari zor diot. Eskerrik asko, hortaz, tesi honetan eginiko ikerkuntzan kolore apur bat sartzen laguntzeagatik.

No puedo dejar sin mencionar el nombre de German Rigau. Todos le conocemos. Sobran las palabras. German, gracias por enseñarme en las charlas que hemos tenido que un ochenta y cinco por ciento no significa nada si no tenemos un sentido global del área de investigación.

I would like to mention Walter Daelemans, who helped me with the central concepts I used in this dissertation. The distinction between robustness and adaptation has been the keystone to keep on with this dissertation.

Ixa taldeari tesia egiteko aukera eman eta lagundu didalako. Bereziki Gorka, Eneko eta Larraitzi depositatzeko orduan emandako laguntzagatik.

Azkenik, korta akademikoan dagoen horri. *Jota agobiauta nau* momentuetan laguntzeagatik. Saiaketan naiz hobeketan, Sam Cooken hitzei eutsiz, *Now I*

don't claim to be an "A" student, But I'm trying to be... eta aurrerago, *What a wonderful world this would be.*

Acronyms and abbreviations

ACL: Association for Computational Linguistics
AI: Artificial Intelligence
BC: Brown Corpus
BNC: British National Corpus
DSO: Defense Science Organization
HLT: Human Language Technology
IE: Information Extraction
IR: Information Retrieval
ITA: Inter-Annotator Agreement
k**-NN**: k Nearest Neighbors
LDOCE: Longman Dictionary of Comteporary English
LKB: Lexical Knowledge Base
LSA: Latent Semantic Analysis
LSI: Latent Semantic Indexing
MFS: Most Frequent Sense
ML: Machine Learning
MLE: Maximum Likelihood Estimation
MT: Machine Translation
NE: Named Entity
NLP: Natural Language Processing
NLU: Natural Language Understanding
OMT: One Matrix per Target word
PPR: Personalized Page Rank
PoS: Part of Speech
SMA: Singular Matrix for All target words
SVD: Singular Value Decomposition
SVM: Support Vector Machines

TC	Text Categorization
UMLS	Unified Medical Language System
VSM	Vector Space Model
VSM	Vector Space Model
WN	WordNet
WSD	Word Sense Disambiguation
WSJ	Wall Street Journal

Contents

CHAPTER I

Introduction

I.1 Word Sense Disambiguation

During this long way, people kept asking me what I was still doing at university. I always answered: *wordsensedisambiguation*[1]. I waited for a few seconds, and then, explained further and tried to define my work. I could say something like, "*You know that words may have different meanings depending on the context in which they occur, right?* [while she or he was nodding] *Well, I'm trying to teach the computer to automatically choose the meaning of a word in text. I'm "teaching" a language*". Afterward, the friend or whoever who had asked the question kept quiet for a while, as if assimilating the information, and then answered: "*Uh! That could be interesting to automatically translate from one language to another!*". This idea is the only feedback I have ever received if there was any answer at all[2].

It is well known that Machine Translation (MT) is more than lexical choice, but these kind of ideas given by people, who is out of our working sphere, make me think that sense ambiguity in text is one of the most natural problems around. And without doubt, one of the essential step in Natural Language Processing (NLP) and Natural Language Understanding (NLU). We do not have to go very far to find an example that shows this

[1]pronounced as a native Basque speaker.

[2]20% of the people gave such idea, the rest is still in an assimilation process or have just gone away.

kind of ambiguities in text. Our life is full of jokes or puns which help to illustrate our problem and aim. For example, if we take a piece of lyric from the Ramones that says:

> *Beat on the brat*
> *Beat on the brat*
> *Beat on the brat with a baseball bat*
> *...*
> *With a brat like this always on your back*
> *What can you do? (lose?)*

Focusing on the word *brat*, every English speaker knows its meaning in the text. If we look it up in a dictionary we would note that a *brat* can be either a small pork sausage or a very troublesome child (more common to be hanged on ones back). By picking up a dictionary it can be seen that a word may has different meanings, some of which are very different. These different meanings of polysemous words are known as senses and the process of deciding which is being to be used in a particular context **Word Sense Disambiguation** (Agirre and Edmonds, 2006, WSD).

WSD, in its broadest sense, can be considered as determining the meaning of every word in context, which appears to be a largely unconscious process in people's minds. As a computational problem it is often described as "AI-complete", that is, a problem whose solution presupposes a solution to complete NLU or common-sense reasoning (Ide and Véronis, 1998).

Computationally, WSD can be seen as a classification problem, where word senses are the classes, the context provides the evidence, and each occurrence of a word is assigned to one or more possible classes based on the evidence. This means that words are assumed to have a finite and discrete set of senses from a dictionary, a lexical knowledge base, or application-specific sense inventories (commonly used in MT). The fixed inventory makes the problem tractable and reduces the complexity of the problem. In terms of evaluation also gives a way to asses and compare each of the methods at stake. Some authors described the limitations of the fixed setting (Killgariff and Tugwell, 2004) and argue that a more dynamic approach should be taken in order to represent the word meaning in a corpus.

Whatever the way we choose to have a deep understanding the language, a robust NLU interface should, in the end, be able to tell which sense, among a list of senses, is intended in a given context.

In the next section we will discuss the problem of word senses and try to give a proper definition of senses for this task. In Section I.3 we will summarize the practical aspects in NLP. Then in Section I.4 we will list the main problems concerning Word Sense Disambiguation. Section I.5 is devoted to motivate the central issue of this dissertation, and Section I.6 presents a technique to mitigate the problems explained in the previous section. In Section I.7 we will describe some related work on domain adaptation, and will locate this dissertation among them. In Section I.8 we will list the contributions of the dissertation. Next, Section I.9 will present a guide to readers, and finally, in Section I.10 we list the publications stemmed from this PhD thesis.

I.2 What is a word sense?

Word meaning is in principle infinitely variable and context sensitive.It does not divide up easily into distinct sub-meanings or senses. Lexicographers frequently observe loose and overlapping word meanings, and standard or conventional meanings which have been extended, modulated, and exploited in a bewildering variety of ways (Killgariff, 1997). In lexical semantics, this phenomenon is often addressed in theories that model sense extension and semantic vagueness, but such theories are at a very early stage in explaining the complexities of word meaning (Cruse, 1986; Tuggy, 1993; Lyons, 1995).

"Polysemy" means to have multiple meanings. It is an intrinsic property of words (in isolation from text), whereas "ambiguity" is a property of text. Whenever there is uncertainty as to the meaning that a speaker or writer intends, there is ambiguity. So, polysemy indicates only potential ambiguity, and context works to remove ambiguity.

At a coarse grain a word often has a small number of senses that are clearly different and probably completely unrelated to each other, usually called *homographs*. Such senses are just "accidentally" collected under the same word string. As one moves to finer-grained distinctions the coarse-grained senses break up into a complex structure of interrelated senses, involving phenomena such as general polysemy, regular polysemy, and metaphorical extension. Thus, most sense distinctions are not as clear as the distinction between *bank* as 'financial institution' and bank as 'river side'. For example, bank as financial institution splits into the following cloud of related senses: the company or institution, the building itself, the counter where money is exchanged, a fund or reserve of money, a money box (*piggy bank*), the funds in a gambling house, the dealer in a gambling house, and a supply of something held in reserve (*blood bank*) (WordNet 2.1).

Given the range of sense distinctions in examples such as these, which represent the norm, one might start to wonder if the very idea of word sense is suspect. Some argue that task-independent senses simply cannot be enumerated in a list (Killgariff, 1997). And perhaps the only tenable position is that a word must have a different meaning in each distinct context in which it occurs. But a strong word-in-context position ignores the intuition that word usages seem to cluster together into coherent sets, which could be called senses, even if the sets cannot be satisfactorily described or labeled.

In practice, the need for a sense inventory has driven WSD research. In the common conception, a sense inventory is an exhaustive and fixed list of the senses of every word of concern in an application. The nature of the sense inventory depends on the application, and the nature of the disambiguation task depends on the inventory. The three Cs of sense inventories are: clarity, consistency, and complete coverage of the range of meaning distinctions that matter. Sense granularity is actually a key consideration: too coarse and some critical senses may be missed, too fine and unnecessary errors may occur. There is evidence that if senses are too fine or unclear, human annotators also have difficulty assigning them.

The "sense inventory" has been the most contentious issue in the WSD community, and it surfaced during the formation of Senseval, which required agreement on a common standard. Various resources have been used (HECTOR, LDOCE, WordNet), which each has each pros and cons (these resources will be reviewed in the following chapter). For example, HECTOR (Atkins, 1993) is lexicographically sound, but lacks coverage; WordNet is an open and very popular resource, but is too fine-grained in many cases. Senseval eventually settled on WordNet, mainly because of its availability and coverage. Of course, this choice sidesteps the greater debate of explicit versus implicit WSD, which brings the challenge that entirely different kinds of inventory would be required for applications such as MT (translation equivalences) and IR (induced clusters of usages). In this dissertation we focus in explicit WSD approach with a fixed list of senses.

I.3 Usefulness of WSD

As Wilks and Stevenson (1996) pointed, WSD is a *intermediate task* (which is not an end in itself) for many other NLP applications and tasks. Machine translation is the original and the most obvious application for WSD. Lexical resolution is essential for many Natural Language Understanding (NLU) tasks such as message understanding and man-machine communication. But not only it is useful for those aims, in some instance WSD has been shown to be useful for NLP. Among others, we could mention the following:

- **Machine Translation**: Sense disambiguation is essential for lexical choice in MT for words that have different translations for different senses. For example, in an English-Basque translator *sister* could be translate to *ahizpa* or *arreba* depending on the genre of the siblings. Most MT systems does not use explicit WSD: Either use pre-disambiguated lexicon for a give domain, hand-crafted rules are devises or WSD if folded in to statistical translation model (Brown *et al.*, 1991). Recent work have shown that WSD is useful to improve MT systems (Carpuat and Wu, 2005; Chan *et al.*, 2007a).

- **Information Retrieval**: WSD is required to remove occurrences in documents where the word or words are used in an inappropriate sense. Thus, given the query "*depression*", what type of documents should retrieve the systems, economic, illness, weather systems? Early experiments suggested that reliable IR would require at least %90 of accuracy in WSD systems. More recently, WSD have been shown to improve cross-lingual IR and document classification (Vossen *et al.*, 2006; Stephan and Hotho, 2004; Clough and Stevenson, 2004; Otegi *et al.*, 2008)

- **Information Extraction and text mining**: Ability to sense distinction is required for intelligent and accurate analysis of text. For example, a gathering system have to only collect those documents about illegal *drugs*, rather than medical *drugs*. The Semantic Web community needs automatic annotation of documents according to a reference ontology. Finally, Named-entity classification can be seen as WSD problem for proper names.

- **Syntactic Parsing**: Sense disambiguation can help in prepositional phrase attachment (Agirre *et al.*, 2008), and in general restricts the spaces of competing parses (Alshawi and Carter, 1994).

- **Lexicography**: WSD and lexicographers can work in a loop. WSD providing rough empirical groupings and statistacally significant contextual indicators of sense to lexicographers, who provide better sense inventories and sense-annotated corpora.

Despite those positive results, WSD has not reached general acceptance, for the following reasons. First, the lexicons used do not fit the domain of the application. Second, WSD might not be accurate enough to show a significant effect. Third, treating WSD as an explicit component means that it cannot be properly integrated into a particular application or appropriately trained on the domain. Research is just beginning on domain-specific WSD, as this dissertation will show.

Nevertheless, we think that applications do require WSD in some form. We think that the work on explicit WSD can serve to explore and highlight the particular features that provide the best evidence for accurate disambiguation, be it implicit or explicit.

I.4 Main difficulties in WSD

Although the long tradition of WSD, the performance of available systems can be seen as moderate. We will dedicate more space to Senseval and SemEval[3] editions, but regarding overall disambiguation performances current systems obtain around 60%-70% of accuracy (Snyder and Palmer, 2004; Pradhan *et al.*, 2007) in all-words disambiguation task using fine-grained senses such as those found in WordNet.

The low results can be explained by the following issues:

- **Sense inventory and granularity**. We reviewed in Section I.2 that the choice of "sense inventory" and the division among senses are still open problems, and the practical definition of sense distinctions with respect to specific applications is not well understood. In the last years WordNet has been widely used as a the sense-inventory in WSD community, and gives the possibility of comparing results of different research groups and fairly assess the advances in the field. However, the sense

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

inventory is clearly too fine-grained for many tasks, and this makes the disambiguation very difficult. As we will see in Section II.4.4 coarser distinction of sense used in SemEval-2007 improved considerably the accuracy of the systems.

- **Feature modeling is too limited**. Big advances have been done in this field during the last years. Features obtained with complex analysis of the text (morphological, syntactic, semantic, domain, etc.) have been added to more traditional set of simple features, such as bigrams, trigrams, and bag-of-words. Such large feature spaces tend to have highly redundant and heterogeneous features, and it is not clear which is the best way to profit from them. We propose to use Singular Value Decomposition. It is also important to note that the interactivity among ML algorithm, parameters, and features types per word should help solve the problem (Hoste *et al.*, 2002).

- **The sparse data problem**. Many current NLP systems rely on linguistic knowledge acquired from text via ML methods. Statistical or alternative models are learned, and then applied to running text. The main problem faced by such systems is the sparse data problem: In NLP most of the events occur rarely, even when large amounts of data are available. This problem in specially noticeable in WSD, due to the small amount of training examples. Only a handful of occurrences with sense tags are available per word. For example, if we take the word *channel*, we see that it occurs 5 times in SemCor (Miller *et al.*, 1993), the only all-words sense-tagged corpus publicly available. The first sense has four occurrences, the second a single occurrence, and the other 5 senses are not represented. For a few words, more extensive training data exists: The Lexical Sample task of Senseval-2 (Edmonds and Cotton, 2001) provides 145 occurrences for *channel*, but still some of the senses are represented by only 3 or 5 occurrences. Therefore, the estimation of rare occurring features is crucial to have high performance, and smoothing techniques, such as Singular Values Decomposition, can be useful in this process. Another possible solution to tackle sparsity problem is the use of unlabeled data and appropriate learning techniques to profit from then, as explored in this dissertation.

- **Portability across domains**. Specific domains pose fresh challenges to WSD systems: different domains involve different predominant senses, some words tend to occur in fewer senses in specific domains, the context of the senses might change, new senses and terms might be involved. Some previous work (Escudero *et al.*, 2000; Martínez and Agirre, 2000) has shown that there is a loss of performance when training on one corpora and testing on another. We propose the use of Singular Value Decomposition to improve portability. As this problem is one of the main topics in this dissertation, we will give further details in the next section.

I.5 WSD across domains

A specific topical domain has its own peculiarities. In a specific domain some words has more relevance than in other domains. These words model the domain, and therefore, some concepts and meanings – those that belong to the domain – are more frequent. That way, in texts which belong to the finance domain words (and respective concepts) such as *stock, company* or *business* are more frequent than in texts from the sports domain. In the latter domain we can expect words like *train, competition* or *goal.* This peculiarities affect directly the performance of WSD systems.

In terms of probabilities, we can say that the priors (relative predominance of the senses with respective to a target word) change when we are moving on different domains – some concepts are more common in specific domains than in others. The distribution of word senses change drastically when we shift the domain. And so do the features, as the relevant words change: Features that are usual in a given domain become rare when we port the WSD system to other domain. These issues difficult the learning for WSD systems which are trained on general domain. In other words, what they learn in one domain is hardly applicable to different domains.

Regarding distribution of sense across domains, we can summarize the following phenomenas:

Word sense distributions might change drastically. For example, in the sports domain the meaning of *coach* is more likely to be *some one to in charge of training an athlete or a team* rather than *a vehicle carrying many passengers or a bus. Score* which is mainly used to mean *the number of points achieved in a game* in the sports domain, but more likely means *a set of twenty numbers* when we read an article from the Financial Times.

This is one of the major problem for supervised WSD systems where the most frequent sense in the training data sets a high bias. The priors in three difference corpus for a handful of nouns are shown in Appendix A. In Chapter V we will show how different sense distribution can affect negatively the performance of the previous WSD systems.

Word sense variability decreases in domain-specific corpora. That is, words tend to occur in less senses, where the probability of the predominant sense is usually higher (high *s*kew) and the use of the rest is lower, and can disappear. The figures in Table I.1 show that in a general domain (BNC corpus) the average number of senses is higher than in finance and sports domains. This fact is interesting for unsupervised systems or approaches which try to find the predominant senses from text. Note that predominant sense of a text might be a powerful baseline, which can be hard to beat for the supervised system (McCarthy *et al.*, 2004). This details will be further explained in Chapter VII.

	#sense avg.	standard dev.
BNC	6.26	1.74
FINANCE	3.87	1.87
SPORTS	3.96	1.73

Table I.1:

New word senses might arise in domain-specific corpora. It is well known that specialized uses of domain words might create a new senses. For instances, *cold* as a noun has five possible meanings in UMLS, which is knowledge base for biomedical purpose, whereas in WordNet (version 3.0) has only three. For this dissertation we do not care about this phenomena, since we have worked with a fixed sense-inventory.

Similar behavior might be expected for learning features. The words in context change across domains, as well as syntax and structures of phrases. Thus, extracted features would be different, and algorithms would generalize differently. In other words, we say that **feature distribution** also changes across domains. This is closely related to the data-sparseness problem in WSD, and it is specially noticeable in WSD, where hand-tagged data is difficult to obtain. Although it does not only belongs to the domain-shift issue, it becomes more critical when domain shift is faced.

I.6 Singular Value Decomposition for data sparseness and domain shifts

In Section I.4 we listed the main problems of WSD. The choice of the sense-inventory was let out of the scope of this dissertation. We focused on the rest of the listed problems: *Feature modeling, data sparseness* and *domain shift*. Our starting point have been to create an effective feature sets to model correctly the sense and tackle the data sparseness and domain shift problems.

Many ML algorithms are based on vector models, where co-occurring features are relevant in order to assign word senses (first-order association). Due to data sparseness and domain shift first-order association might not be enough to learn properly the classification task, and higher order association might be used.

The use of the **Singular Value Decomposition** (SVD) and unlabeled data might be helpful to mitigate the data sparseness problem, and make possible to port WSD system across domains. SVD finds a condensed representation and reduce significantly the dimensionality of the feature space (all these details will be further explained in Chapter III). This re-representation captures indirect, high-order associations. As Landauer and Dumais (1997) explained, if a particular stimulus, $\mathcal{X}$, (e.g., a word) has been associated with some other stimulus, $\mathcal{Y}$, by being frequently found in joint context (i.e., contiguity), and $\mathcal{Y}$ is associated with $\mathcal{Z}$, then the condensation can cause $\mathcal{X}$ and $\mathcal{Z}$ to have similar representations. However, the strength of the indirect $\mathcal{X}\mathcal{Z}$ association depends on much more than a combination of the strengths of $\mathcal{X}\mathcal{Y}$ and $\mathcal{Y}\mathcal{Z}$. This is because the relation between $\mathcal{X}$ and $\mathcal{Z}$ also depends, in a well-specified manner, on the relation of each of the stimuli, $\mathcal{X}$, $\mathcal{Y}$, and $\mathcal{Z}$, to every other entity in the space.

It is said that SVD helps to improve the similarity measure in three ways:

1. **High-order co-occurrence**: Dimension reduction with SVD is sensitive to high-order co-occurrence information. It takes into account the indirect associations that improve the similarity measures.

2. **Latent meaning**: SVD creates a low-dimensional linear mapping between words or features and chunks of text (whatever they are). This

mapping captures the latent (hidden) meaning in the words and the chunks.

3. **Noise reduction**: Dimension reduction removes random noise from the input matrix (words or features by chunk of text), and we can say it *smooths* the matrix. The raw matrix contains a mixture of signal and noise. A low-dimensional linear model captures the signal and reduces the noise.

Alternatively, there is also the preoccupation about the best way to apply ML techniques to supervised settings. The Senseval-3 exercises showed that the more feature types one throws into the algorithm, the better are the results (Agirre and Martínez, 2004a). Still, it is not clear which is the best way to profit from the very rich feature space. Apart from the sparsity problem already mentioned, large feature spaces tend to have highly redundant and heterogeneous features. As a potential solution, we interpret that SVD collapses similar features (i.e. having similar distributions), and will thus be helpful against sparsity, heterogeneity and redundancy.

As mentioned in previous sections, domain shifts intensify the sparseness of the data, due to the fact that the context of sense occurrences are different, making it more difficult to find overlap among train and test occurrences. For instance, let's assume two instances for the target word *b*ank , one from a training set obtained from an open domain corpus (for instance, the British National Corpus (Leech, 1992)) and the other from a domain-specific testing set about sports (for example from sports section of the Reuters corpus (Rose *et al.*, 2002)):

Train$_1$ $\Rightarrow$ *Each fish is caught in the water, then carried to nearby* **bank#1** *of the river and delicately devoured.*

Test$_2$ $\Rightarrow$ *The winner sailing ship crossed the finishing line while tens of thousands of cheering people lined the* **banks#?**.

In the above examples, the overlap between the context of both train and test occurrences of bank is null. It thus becomes much harder to assign the correct sense, unless we find a method to tell us whether those occurrence contexts are similar. A solution we consider is to calculate higher-order correlation among the elements in the contexts, and thus make possible to measure some kind of similarity.

Basically our idea is twofold: use SVD in order to find indirect association and measure better similarities, and use unlabeled data in order to help finding better feature correlation.

	train$_1$	test$_2$	unl$_3$	unl$_4$	unl$_5$
winner	0	1	0	1	0
ship	0	1	1	0	0
line	0	1	0	0	0
people	0	1	0	0	1
fish	1	0	0	0	1
water	1	0	1	0	0
river	1	0	1	0	0
investor	0	0	0	1	1
finance	0	0	0	1	0

Table I.2: Term-by-instance matrix showing that there is no any co-occurrence between train$_1$ and test$_2$, but the use of unlabeled data can help finding indirect association (unl$_3$)

Table I.2 shows a term-by-instances matrix for the example above. Rows represent the words occurring in different text-chunks, and columns represent the instances or text-chunks. This matrix illustrates the example above, and shows how train$_1$ and test$_2$ instances do not have context in common, but how using unlabeled data it is possible to create indirect associations. *Water* and *ship* are closely related, and thus they tend to occur in the same context, as attested in the unlabeled instance unl$_3$.

SVD reduces the space and finds correlations collapsing features, thus associating semantically related feature and collapsing such features onto the same feature in the new representation. This way, we can calculate in better way similarities among occurrences (text instances).

Table I.3 shows the instances represented in a reduced space obtained applying SVD to the matrix in Table I.2. SVD makes explicit the relation between some words in train$_1$ and test$_2$ via indirect associations created with unlabeled data.

SVD provides a principled method to alleviate sparse data problems and domain shifts, but it does not specifically address changes in sense distribution. We will give more details and examples on the use of SVD in Chapter III.

	train_1	test_2	unl_3	unl_4	unl_5
dim_1	-0.70	-0.26	-0.97	-0.44	-1.62
dim_2	0.35	0.65	1.00	-0.84	-0.46

Table I.3: Instances represented in the 2-dimensional reduced space. train_1 and test_2 are now highly correlated, as they have similar weights in the new space.

I.7 Prior work in domain adaptation

Since the domain adaptation problem has a central role in this dissertation, we will now focus on relevant literature. We will focus on machine learning systems, the most active field in this area. The existing works are categorized based on how the connection between the source domain and target domain is modeled. We refer to training domain as source domain whereas test domain, where labeled data is very little, as target domain.

Instance weighting. These models use Empirical Risk Minimization for standard supervised learning (Vapnik, 1995) in order to derive an instance weighting solution to domain adaptation. They assign a instance-dependent weight to the loss function when minimizing the expected loss over the distribution of the data. In this category we can find two types of assumption over the data distribution.

On one hand, we have the assumption that the distribution on the source and target domains are the same in the given same class (word sense). However, the class distribution may be different. Thus, we only need to weight the instances with the difference of the class probability in each domain. In these terms, Zhu and Hovy (2007) use resampling techniques in active learning for word sense disambiguation, over-sampling under-represented word senses, and under-sampling over-represented senses. Chan and Ng (2005), also for WSD try to model the target probability distribution transforming the source distribution, modeling directly with a logistic regression classifier. They use the Expectation-Maximization (EM) algorithm to estimate the class distribution in the target domain. In (Chan and Ng, 2006), they perform a similar trick but applied on a Naive Bayes algorithm for WSD.

On the other hand, one can make the assumption that when the observation of the input variables (feature distribution) is the same, the conditional distribution of the classes (word senses) on each domain are the same. But the marginal distribution of the features can be different in both domains. This could be problematic when misspecified models are used, since the error is specially minimized in dense regions of source domain distribution (training data), which have different dense region to target domain. Thus, the induced model will no be longer optimal for the target domain. Shimodaira (2000) proposed to re-weight the log likelihood of each training instances using the ratio between marginal distribution of features of each domain in maximum likelihood estimation for covariate shift.

Semi-supervised learning. Although this category is not only related to the domain adaptation problem, these kind of methods can be readily deployed in this problem. Using unlabeled data from the target domain (assuming it is easy to obtain) we can apply any semi-supervised learning algorithm to the domain adaptation problem (Zhu, 2005; Chapellea *et al.*, 2006). Dai *et al.* (2007b) proposed an EM-based algorithm for domain adaptation. They estimate the trade-off parameter between labeled (source) and unlabeled (target) data using KL-divergence. Jiang and Zhai (2007a) proposed to not only include weighted source domain instances but also weighted unlabeled target domain instances in training using bootstrapping, which essentially combines instance weighting with bootstrapping. In (Xing *et al.*, 2007), the authors try to make the domain adaptation based on graph algorithms and the use of the unlabeled data from the target domain. Although some works do not address directly the domain adaptation problem, there are couple of works that make use of unlabeled data and interestingly could be applied in the domain adaptation problem: Pang and Lee (2004) with Min-Cut algorithm or Label propagation on k-NN graph (Niu *et al.*, 2005). Our work in this dissertation can be considered as semi-supervised learning, since we use unlabeled data in order to improve the performance in the target domain.

Change of feature representation. Another way to perform the adaptation to the new domain is to change the representation of the instances and thus avoid the problem of the differences between distributions in the source and target domains. The solutions proposed assume that it is possible to find a new representations of the features where the joint probabilities in source and target domain are the same.

A special and simple kind of transformation is feature subset selection.

Satpal and Sarawagi (2007) proposed a feature subset selection method for domain adaptation, where the criterion for selecting features is to minimize an approximated distance function between the distributions in the two domains. Blitzer *et al.* (2006) proposed a structural correspondence learning (SCL) algorithm that makes use of the unlabeled data from the target domain to find a low-rank representation that is suitable for domain adaptation. It is empirically shown in (Ben-David *et al.*, 2007) that the low-rank representation found by SCL indeed decreases the distance between the distributions in the two domains. The core algorithm in SCL is from (Ando and Zhang, 2005). Ben-David *et al.* (2007) formally analyzed the effect of representation change for domain adaptation. They proved a generalization bound for domain adaptation that is dependent on the distance between the induced joint probability distribution given by the new representation in both domains.

Our main motivation explained in previous section fits in this categorization. Using SVD we find that the features that manage the similar information in two domains are collapsed in the same feature and this way the distance of the source and target domain can be minimized.

Multi-task learning is highly related to domain adaptation. In multi-task learning there is single distribution of the observation and a number of different output variables. We can cast the domain adaptation as multi-task formulating two tasks, one for each domain. Indeed, Daumé III (2007) proposed a simple method for domain adaptation based on feature duplications (cf. Section III.2.1.6). The idea is to make a domain-specific copy of the original features for each domain (training data consist of examples from source and target domains). An instance from domain k is then represented by both the original features and the features specific to domain k. Similarly, Jiang and Zhai (2007b) proposed a two-stage domain adaptation method, where in the first generalization stage, labeled instances from k different source training domains are used together to train k different models, but these models share a common component, and this common model component only applies to a subset of features that are considered generalizable across domains.

Ensemble Methods. Another set of models that could be useful for domain adaptation are those which construct complex models using ensembles of classifier. Among these, we can find bagging, boosting or mixture of models. For instance, a mixture model proposed by Daumé III and Marcu (2006) for domain adaptation, in which three mixture components are assumed, one shared by both the source and the target domains, one specific to the source domain, and one specific to the target domain. Labeled data from both the source and the target domains is needed to learn this three-component mixture model using the conditional expectation maximization algorithm. Storkey and Sugiyama (2007) considered a more general mixture model in which the source and the target domains share more than one mixture components. Regarding Boosting methods, Dai *et al.* (2007a) proposed to modify the widely-used AdaBoost algorithm to address the domain adaptation problem. The idea here is to put more weight on mistakenly classified target domain instances but less weight on mistakenly classified source domain instances in each iteration.

I.8 Contribution of the dissertation

Our aims for this dissertation are twofold. First, we want to shed light on the sparse-data problem and large sets of redundant and heterogeneous features spaces, as used in WSD. Second, we would like to explore domain adaptation for WSD systems. Our main hypothesis is the following:

> The use of Singular Value Decomposition (SVD) can provide a better representation for machine-learning Word Sense Disambiguation.

We propose different scenarios to explore the above hypothesis, developing approaches not previously described in the literature. All in all, we think that our contributions on this initial hypothesis are the following (including their relation to chapters in the dissertation):

- **SVD decomposition is useful to deal with data sparseness and domain adaptation** (Chapters 4, 5 and 6): We explored the contribution of features obtained with SVD decomposition to WSD performance. We presented several experiments across different datasets. We studied the performance of a number of ML algorithms trained on these types of features and analyzed the effect of the number of dimensions. We also developed different ways to obtain the SVD representation, each catching different evidences from text. The SVD representation is complementary to the *classic* feature set, and we showed that combining them is a robust way to improve the results. We used two experimental scenarios: general domain WSD was tested in Senseval and SemEval datasets, and domain-specific WSD. Our results obtain the state-of-the-art over Senseval-like datasets. In domain adaptation, we showed that SVD features are good enough to obtain robustness (on a general WSD system) and adaptation (on a system trained with examples from general domain and domain-specific instances).

- **Unlabeled data helps find better correlations among features** (Chapters 4, 5 and 6): We studied the usefulness of unlabeled data to obtain better correlation among the features from labeled instances. We use unlabeled data to help find higher-order correlations applying SVD to the augmented matrix. In order to asses the effect of the unlabeled data we evaluated WSD systems trained on different amounts of unlabeled data. We reported several experiments showing that unlabeled data help up to certain amounts. On the domain adaptation scenario, we played with unlabeled data from different sources, finding that unlabeled data must be topically related to the training set in order to obtain effective SVD features. We showed that unlabeled data helps obtain more reliable features and an it is an important factor in the domain adaptation scenario.

- **Combination of feature spaces is useful** (Chapters 4 and 6): the redundancy and heterogeneity of features can affect negatively. We split the original feature set in coherent sets, and explored how to combine them. In k-NN combination each k-NN system is trained on a different feature space and casts votes for its first k neighbors. In kernel-based combination, each kernel's implicit function is a different kind of SVD mapping. The former method obtain the state-of-the-art results in Senseval dataset. The latter showed that it is an effective way to take profit of the general source domain in the supervised domain adaptation scenario. We show that a combination of rich feature spaces is a useful and robust manner to obtain good results for WSD.

- **Robustness in face of semi-supervised domain adaptation** (Chapter 5). Using SVD and unlabeled data we obtained a robust system that performs well across different target domains without labeled data from the target domains, reducing the domain shift problem for general WSD system. We found that in order to obtain a robust system the unlabeled data should be from general domain and be related to the training set.

- **Supervised domain adaptation** (Chapter 6). We have showed for the first time that source general domain examples are an useful addition to target domain examples in WSD, and provide for the best results domain adaptation. Up to now, the general domain examples were not shown to be useful We concluded that the correlations found by SVD and the generalization provided by the combination of SVD-based kernels are effective.

In order to make a more complete picture of domain adaptation, we also explored the performance of knowledge-based models:

- **Knowledge-based WSD system may outperform a general WSD system** (Chapter 7): We explored the application of knowledge-based WSD systems to specific domains, based on a combination of state-of-the-art graph-based WSD system (Agirre and Soroa, 2009) that uses the information in WordNet with a distributional thesaurus built from the target domains. This system outperformed supervised systems trained on SemCor, showing that knowledge-based WSD systems are a powerful alternative to supervised WSD systems.

I.9 Structure of the dissertation

- **First chapter** – *Introduction*
- **Second chapter** – *State-of-the-art: resources, systems and evaluation.* This chapter is devoted to the description of different methods and research lines that are presenting promising results in the WSD task. It describes the main resources that are employed for WSD, including lexical databases, corpora, and some well-known learning algorithms. An important section will be dedicated to the Senseval competitions and the participating systems. The last section will describe the-state-of-art of domain-specific WSD,including available resources and different settings.
- **Third chapter** – *Methods and approaches from this dissertation.* In this chapter we describe the methods used in the whole dissertation. It

includes the feature types used in supervised WSD, the foundations of Singular Values Decomposition, SVD features, and the combination of k nearest neighbor (k-NN) systems and kernel-based combinations.

- **Fourth chapter** – *Combination of feature spaces and unlabeled data with* SVD *for* WSD. In this chapter we analyze the contribution of SVD, the use of unlabeled data, and feature split for general WSD. We apply k-NN combination to use different feature spaces, and obtain better performances. We will rely on different ML methods to study the contribution of the new features, tested on Senseval-2, Senseval-3 and SemEval-2007 lexical-sample and all-words tasks.

- **Fifth chapter** – *Semi-supervised domain adaptation.* In this chapter we explore robustness and adaptation issues for WSD using SVD and unlabeled data. We focus on the semi-supervised domain adaptation scenario, where we train on the source (general domain) corpus and test on the target (domain-specific) corpus, and try to improve results using unlabeled data. We study the importance of the domain source of the unlabeled data in order to obtain effective features.

- **Sixth chapter** – *Supervised domain adaptation.* The goal of this chapter is to adapt a WSD system in a supervised domain adaptation scenario, where we use both source and target corpora for training, while we test on target corpora. We rely on SVD, unlabeled data and kernel and k-NN combinations. In addition, we study how many target corpus example we need to get the adaptation to the target domain.

- **Seventh chapter** – *Unsupervised domain-specific* WSD. This chapter explores the application of knowledge-based word sense disambiguation systems to specific domains. We focus on graph-based WSD that uses information in WordNet combined with distributional thesauri. We compare the performance to state-of-the-art supervised systems. We also analyze the importance of the predominant sense in specific-domain corpora.

- **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.

I.10 Publications related to this dissertation

In this section we list all our publications related to this thesis, organizing them according to the dissertation chapters.

Chapter IV:

- Agirre, E., Lopez de Lacalle, O. and Martínez, D. Exploring feature spaces with SVD and unlabeled data for Word Sense Disambiguation. In *Proceedings of the Conference on Recent Advances on Natural Language Processing* (RANLP-05). Borovetz, Bulgaria, 2005.

- Agirre, E., Lopez de Lacalle, O. and Martínez, D. Exploring feature set combinations for WSD. In *Proceedings of the XXII Conference of Sociedad Espaola para el Procesamiento del Lenguaje Natural* (SEPLN-06). Zaragoza, Spain, 2006.

- Agirre, E. and Lopez de Lacalle, 0. UBC-ALM: Combining k-NN with SVD for WSD. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Association for Computational Linguistics.* Prague, Czech Republic, 2007.

Chapter V:

- Agirre, E. and Lopez de Lacalle, O. On Robustness and Domain Adaptation using SVD for Word Sense Disambiguation. In *The 22nd International Conference on Computational Linguistics*, (COLING-08). Manchester, UK, 2008.

Chapter VI:

- Agirre, E. and Lopez de Lacalle, O. Supervised Domain Adaption for WSD. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, (EACL-09). Athens, Greece, 2009.

Chapter VII:

- Agirre, E., Lopez de Lacalle, O. and Soroa, A. Knowledge-based WSD on Specific Domains: Performing better than Generic Supervised WSD. In *Proceedings of the Twenty-First International Joint Conferences on Artificial Intelligence* (IJCAI-09). Pasadena, California, USA, 2009.

CHAPTER II

State-of-the-art: resources, systems and evaluation

In this chapter we will present the state-of-the-art for WSD. This task has received a great deal attention from many researches in NLP during the years. Because an extensive survey of these works is out of scope of this dissertation, we will organize the chapter as follows. First we will briefly introduce the main resources that are applied to WSD research: Lexical databases and publicly available corpora. Next we will describe which is the usual way to encode knowledge for WSD. The next section will be devoted to a classification of well-known algorithms used for WSD. After that, we will focus on the evaluation of WSD systems: measures, significance tests, and the last three Senseval/SemEval competitions. The last section will be dedicated to introduce the state-of-the-art of domain-specific WSD.

II.1 Resources

This section is devoted to the main resources for WSD. First we will list the mostly used lexical databases or dictionaries, although not all the listed dictionaries have been used in the experiments. Next, the most important corpora for English will be introduced. Both hand-tagged and raw corpora will be described.

II.1.1 Lexical databases and dictionaries

As previously said, in this section will introduce the main lexical repositories used for WSD. Not all the repositories described have been used in this dissertation, although, they have been widely used in previous work. For further details see Apendix A in (Agirre and Edmonds, 2006).

Among all the lexical repositories described bellow we mainly have used WordNet, which has played the central role as sense-invento-ry during the whole dissertation, although in some experiments (cf. Section IV.5) we used OntoNotes.

Roget's Thesaurus

The older 1911 edition has been made freely available by Project Gutenberg. Although it lacks many new terms, it has been used to derive a number of knowledge bases, including Factotum. A newer edition (*Roget's Thesaurus of English Words and Phrases*) contains over 250,000 entries arranged in 6 classes and 990 categories.

WordNet lexical database

Princeton English WordNet (Fellbaum, 1998) is a lexical database developed at Princeton University[1]. 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* }[2] .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

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

[2]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
SemeEval-2007 all-words (task 17)	2.1

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.

other languages). The current version (March, 2009) is 3.0 . 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.1.2.

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).

Unified Medical Language System

The UMLS is composed of several knowledge sources, including the Metathesaurus is a large, multipurpose and multilingual vocabulary database which contains information about biomedical and health-related concepts, their various names and the relations among them. The current release contains 135 semantic types and 54 relationships.

II.1.2 Corpora

In this dissertation we have used both labeled and unlabeled corpora. Hand-tagged tagged corpora have been used to train and assess our contribution to WSD. On the other hand, unlabeled data have been used for semi-supervised learning, specially in domain-adaptation experiments.

Although we do not used all the listed corpora below, we decided to describe them all in order to have bigger scope of the resources in WSD. First, we will describe the unlabeled corpora, and next we will focus on hand-tagged corpora. The corpora used in this dissertation are the following:

- Unlabeled corpora: In order to improve the performance of WSD systems and experiment on domain adaptation we have use the British National Corpus (BNC) and The Reuters News Corpus, the sports and finance subsets.

- Sense-tagged corpora: The preliminary experiments have been carried out on Senseval-2, Senseval-3 and SemEval-2007 lexical-sample and all-words task datasets. SemCor corpus has been used as training corpus for all-words tasks. For domain adaptation experiments we used the Domain-specific Sussex corpus.

II.1.2.1 Unlabeled corpora

British National Corpus

The British National Corpus (BNC) has been built as a reasonably balanced corpus: for written sources, samples of 45,000 words have been taken from various parts of single-author texts. Shorter parts (up to 45,000 words) as a multi-author texts (magazines, newspaper and so on) were in full, avoiding over-representing idiosyncratic texts. In total, BNC is a 100 million word collection of samples of written and spoken language.

Brown Corpus

The Brown Corpus (BC) is a million-word "balanced" collection of texts published in United States in 1961. It contains samples of varied written prose: press articles (news, reviews, and reportage), fragments of scientific texts, and fiction, among other categories. The 500 documents (2000 word-long each) are classified into 15 categories.

The Reuters News Corpus

This Corpus has been widely used in NLP, especially in documents categorization. There is a specialized hand tagged corpus useful for domain-specific research (see below). The Sports and Financial parts of Reuters have been used for this dissertation.

II.1.2.2 Sense-tagged corpora

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 PoS, lemma and 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). Although the original SemCor was annotated with WordNet version 1.6, the annotation have been automatically mapped into newer versions. More details in Chapter IV, where we used it as training corpus. It is also used and mentioned in Chapter VII.

Senseval and SemEval test Suites

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 (Atkins, 1993), and their polysemy ranges from 2 to 15 senses. The examples are extracted from a pilot of the BNC.

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. 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.

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 TreeBank II. The sense inventory used for tagging is the WordNet 1.7 pre-release. All content words are sense-tagged, including multi-word constructions.

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*[3] 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.

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).

SemEval-2007 English lexical-sample corpus. This task consists of lexical sample style training and testing data for 35 nouns and 65 verbs in the WSJ Penn Treebank II as well as the Brown corpus. This data will include, for each target item: OntoNotes sense tags (these are groupings of WordNet senses that are more coarse-grained than traditional WN entries, and which have achieved on average 90% ITA), as well as the sense inventory for these lemmas. The training set consist of 22,281 instances and there were 4.851 instances for testing.

SemEval-2007 English all-words corpus. We have supplied a 5000 word chunk of WSJ where all of the verbs and the head words of the verb arguments have WordNet 2.1 sense tags. This is for testing purposes only, and has no training annotation associated with it, or PropBank or VerbNet labels. Overall, only

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

465 words were tagged in the test set.

Defense Science Organisation corpus

This corpus was compiled by the team at the Defense Science Organisation (DSO) of Singapore (Ng and Lee, 1996), which comprises sentences from two different corpora. The first is the Wall Street Journal (WSJ), which belongs to the financial domain, and the second is the Brown Corpus (BC) which is a general corpora of English usage. 191 polysemous words (nouns and verbs) of high frequency in WSJ and BC were selected and a total of 192,800 occurrences of these words were tagged with WordNet 1.5 senses, more than 1,000 instances per word in average. The examples from BC comprise 78,080 occurrences of word senses, and examples from WSJ consist on 114,794 occurrences. In domain adaptation experiments, the Brown Corpus examples play the role of general corpora, and the examples form the WSJ play the role of domain-specific examples. It have been widely used in domain adaptation experiments.

Domain-Specific Sussex corpus

This sense tagged corpus was first reported in (Koeling *et al.*, 2005). The examples come from the BNC (Leech, 1992) and the sports and finance sections of the Reuters corpus (Rose *et al.*, 2002), comprising around 300 examples (roughly 100 from each of those corpora) for each of the 41 nouns. The nouns were selected according to the following criteria: 18 words having at least one synset labeled as Sports or Finance according to WordNet Domains, 8 words which had salient frequencies in each domain (according to the normalized document frequency), and 7 words with equal salience in both domains. The occurrences were hand-tagged with the senses from WordNet version 1.7.1 (Fellbaum, 1998). In domain adaptation experiments the BNC examples play the role of general corpora, and the finance and sports examples the role of two specific domain corpora. We have used as dataset in domain adaptation experiments described in chapters V, VI and VII.

National Library of Medicine WSD Test Collection

The NLM WSD Test Collection, a dataset for biomedicine, was developed by Weber *et al.* (2001), and has been used as a benchmark by many independent groups. The UMLS Metathesaurus was used to provide a set of possible meanings for terms in biomedical text. 50 ambiguous terms which occur frequently in MEDLINE were chosen for inclusion in the test set. 100 instances of each term were selected from citations added to the MEDLINE database in 1998 and manually disambiguated by 11 annotators. Twelve terms were flagged as "problematic" due to substantial disagreement between the annotators. In addition to the meanings defined in UMLS, annotators had the option of assigning a special tag ("none") when none of the UMLS meanings seemed appropriate.

II.2 Features for WSD

Word Sense Disambiguation researchers have identified a wide range of linguistic phenomena, such as selectional preferences and domain information, relevant to resolving word sense ambiguity. Such linguistic phenomena are known as *knowledge sources*. According to Stevenson and Wilks (2001), there are three main classes of knowledge sources: syntactic, such as grammatical structures of sentences, part-of-speech tags and subcategorization information; semantic, for example, selectional preferences and associations between word meanings; and pragmatic/topical, which relate to the role of the word within the wider discourse, such as the information about the topic of the text.

However these knowledge sources are relevant to the disambiguation of words, they need to be coded as features. Features can be defined as different ways to encode in an algorithmic level such sources. For instance, the domain of a word sense can represented by the words co-occurring often with the word sense (bag-of-words feature) as extracted from sense-tagged corpora, or the domain code assigned to the word sense in a specific machine-readable dictionary (MRD) or lexical knowledge base (LKB). Encoding features are extracted from specific resources, as bag-of-words feature from sense-tagged corpora, which, in this case, is the specific resource.

We now present a list of the features which are most commonly used in WSD systems. Following each feature is a description of the knowledge source and lexical source from which it may obtained. Depending on the type of knowledge source that extracts the feature we can categorize as **target-word specific feature**[4], **local feature** or **global feature** (further details

[4]"Target word" refers to the word being disambiguated.

in (Agirre and Edmonds, 2006)). Refer to Section III.1 to see the specific learning features used in this dissertation.

Target-word specific features

- *Word Form of Target Word*: This feature may partially encode part-of-speech (PoS) and morphology, depending on the language.

- *Part of Speech of Target Word*: A direct encoding of the PoS. It is available in lexicons commonly used for WSD and these can be used to determine the grammatical category of the senses.

- *Sense Distribution of the Target Word*: Encodes the frequency of the senses. In principle this distribution could be calculate form sense-tagged corpora, but this would suffer from data sparseness problems unless the corpus was extremely large. Unfortunately, no appropriate resource is currently available and sense distributions are domain dependent, which makes still more difficult to found feasible distribution.

Local features

- *Local Patterns*: These are some of the most commonly used features in WSD, which are based on local patterns around the target word. The local patterns around the target word have many potential ways to encode the knowledge sources. A very common extent of patterns include n-grams around the target word, nth word to the right and left to the target word and so on. Several features in the context could be used to fill this patterns such as word form in text, word lemmas, their PoS, or a mixture of these. These features are most easily extracted from tagged corpus. In an untagged corpus it is difficult to say which of word senses the pattern apply to.

- *Subcategorization*: The details of word's subcategorization behavior are usually extracted from tagged corpora using robust parser. For example, the verb *to grow* is intrasitive when it is used in the "become bigger" sense (*john grew quickly*) but transitive in all other senses (*John grew the plants*).

- *Syntactic Dependencies*: This feature encodes syntagmatic relations, which the dependencies for word sense can be extracted from a parsed and sense-tagged corpus.

- *Selectional Preferences*: Some of the lexicon include this information, for example those used by Wilks (1978) and McRoy (1992). MRDs often include selectional preference information. For example, LDOCE has this information based on a set of 36 semantic types that Stevenson (2003) used for WSD.

Global features

- *Bag-of-Words*: This feature encodes the semantic and topical word associations, as well as domain information. The features consist of a list of words occurring in a wide context window around the target word and how many the occur. More complex features may be formed as bag-of-bigrams. Although no linguistic processing is required, the most of the WSD systems used the lemmatized form the bag-of-words.

- *Relation to Word in Context*: As bag-of-words this feature encodes the semantic and topical word association, as well as domain information, but it has been usually extracted from dictionaries. Lesk (1986) suggested that counting the number of content words shared by definition of two sense provide a measure of semantic relatedness. This is possible to combine the evidence given by the bag-of-words with the dictionaries.

- *Similarity to Words in Context*: Encodes paradigmatic relations. Taxonomic information (WordNet) can be used to find similarity between the senses of the target word and words in the context.

- *Domain Codes*: Encodes the domain information. Some lexical resources list the most likely domain code for each sense. WordNet Domains (Magnini and Cavagliá, 2000) links each sense with a set of domain codes. This resource provide a set of 200 domain tags hierarchically structured. For example, the first sense for *bank*, the sense of financial institution, is attached to the *economy* domain tag, while the second sense, the sloping land sense, is subject to *geography* and *geology* domain tags. In the case of LDCOE, it uses a set of 100 subject codes and 246 subdivision. Finally, the *Roget's International Thesaurus* (Chapman, 1977) also includes domain codes in the form of the categories into which the words are organized.

II.3 WSD Algorithms

The most common WSD approach categorization is based on which information source they use in order to solve the knowledge acquisition bottleneck (Stevenson, 2003).

1. **Knowledge based approaches**
   These approaches are based on a lexicon or knowledge base. The make use of the explicit information gathered from the lexicon. The lexicon may be a machine readable dictionary or thesaurus such as LDOCE or WordNet (described in Section II.1.1) or it can be hand-crafted.
   These approaches are said to be unsupervised systems, since they do not use any sense-tagged corpora (see below the explanation of this). Some of this systems are described in Section II.3.1.

2. **Corpus based approaches**
   These methods perform WSD using information gained form training on some corpora. The corpora can be tagged or untagged and, thus, the approaches which belong to this category can be further sub-classified:

   (a) *Tagged corpora*
   The corpora which is used is previously semantically disambiguated. Some of this kind of corpora are described in Section II.1.2.2. The algorithms that fall in this category are considered supervised, and some the approaches are described in Section II.3.2.

   (b) *Untagged corpora*
   The information is gathered from raw corpora, which have not been semantically tagged. These approaches are taken as unsupervised methods.

3. **Hybrid and semi-supervised learning approaches**
   These are approaches which use a mixture of corpus data and knowledge from a explicit knowledge base. Depending on if they use tagged

corpora or not will be considered supervised or unsupervised systems. Nowadays most of the unsupervised approaches fall in this category. Some relevant approaches are described in Section II.3.1.2.

We also include in this category methods that use both labeled and unlabeled data in order to improve the performance. This dissertation uses unlabeled corpora in order get better performance in general domain and domain-specific corpora, as described in Chapter IV, and Chapter V and Chapter V, respectively.

Nowadays the best performing systems try to use all the information available. Basically, the assignment of the word senses is accomplished by using two major sources of information:

- The context of the word to be disambiguated.
- External-knowledge sources such as lexical, encyclopedic dictionaries, as well as hand-devised knowledge sources, which provide data useful to associate words with meanings.

Traditionally, the unsupervised methods are those which do not assign any fixed word sense from a sense-inventory, but rather make distinctions in meaning based on *distributional similarity* or *translational equivalence*. The former distributional approaches make distinction in word meanings based on the assumption that words that occur in similar context will have similar meanings (Harris, 1968; Miller and Charles, 1991). The latter are based on parallel corpora, which identify translations of a word to target language that are dependent on the sense of the word in source language.

In the strict sense, unsupervised methods are not guided by handcrafted examples or knowledge resources (e.g., WordNet). However, "unsupervised" has become a polysemous term in the word sense disambiguation literature. Thus, in order to avoid any confusion we will call unsupervised method those which are "not supervised", including any method that does not use supervised learning from sense-tagged text, even if they make use of a knowledge base.

So we will only distinct among the approaches in the meaning of the use of hand-tagged data. Those that rely on sense-tagged data are considered as **supervised** approaches; otherwise they are considered **unsupervised**.

II.3.1 Unsupervised WSD techniques

Unsupervised learning is the greatest challenge for WSD researchers. The underlying assumption is that similar senses occur in similar contexts, and thus senses can be induced from text by clustering word occurrences using some measure of similarity of context. Then, new occurrences of the word can be classified into the closest induced cluster/senses. This is called *sense induction* (Schütze, 1998).

Evaluation of sense induction techniques is not straightforward (Schütze, 1998).The performance is below supervised systems (cf. Section II.3.2) and they can hardly beat the Most Frequent Sense (MFS) baseline (cf. Section III.2.1.1). Focusing on the results obtained in the all-words task of the different Senseval and SemEval editions (cf. Section II.4) we can conclude that the best systems' performance is between 53% and 58% of F-score. In the best cases some systems will slightly outperform the MFS baseline.

Next we will describe the most important methods: those which are entirely based on a lexical knowledge base, and those which distributional similarity is an important component.

II.3.1.1 Unsupervised lexical knowledge based methods

These methods use the information in a lexicon or knowledge base in order to perform disambiguation.

The Lesk algorithm is a classical algorithm for word sense disambiguation introduced by Lesk (1986). It is based in the distributional similarity assumption. Given a word to disambiguate, the dictionary definition or gloss of each of its sense is compared to the glosses (or definition) of every other word in the context. A sense whose gloss shares the largest number of words in common with the glosses of the words in context is assigned.

Other methods rely on the explicit structure of knowledge bases. For instance, some algorithms for WSD rely on selectional preferences as a way of constraint the possible meaning of a word in a given context. Selectional preferences capture information about the possible relations between word categories, and represent common sense knowledge about classes and concepts. For instance, EAT-FOOD, DRINK-LIQUID are examples of such semantic constraints. This constraints may be uses as a rule to select the correct word sense. Several approaches have been proposed to acquire and determine the selectional preference between two concepts. Brockmann and Lapata

(2003) give a detailed analysis of these approaches, while Agirre and Martínez (2001b) report a comparative evaluation of some of these approaches.

Finally, there is a number of methods which are based on semantic similarity or relatedness. The idea is that knowledge bases provide information to determine semantic similarity, and it's thus possible to select the sense of the target word which is most similar to the words in context. Several semantic similarity measures have been proposed in the literature, all of them computing metrics on semantic nets. Some of them calculate the similarity as the minimum length between concepts (Leacock *et al.*, 1998; Hirst and St-Onge, 1998). Resnik (1995) defines the notion of information content (the probability of occurrence in a large corpus), and defines a measure of semantic relatedness between words by quantifying the information content of the lowest common subsumer of two concepts. Mihalcea and Moldovan (1999) introduce a formula to measure similarity between words (including for different PoS) creating connections through the glosses. Agirre and Rigau (1996) introduce the notion of conceptual density, defined as the overlap between the semantic hierarchy rooted by a given concept C, and the words in the context of C.

Recently, graph-based methods for knowledge based WSD have gained much attention in the NLP community (Sinha and Mihalcea, 2007; Navigli and Lapata, 2007; Mihalcea, 2005; Agirre and Soroa, 2008, 2009). These methods use well-known graph based techniques to find and exploit the structural properties of the graph underlying a particular knowledge base. Graph based WSD methods are particularly suited for disambiguating word sequences, and they manage to exploit the interrelations among the senses in the given context.

We will focus later on **Personalized PageRank** approach for WSD introduce in Agirre and Soroa (2009), since it has been our unsupervised method of choice for the domain-specific WSD experiments reported in Chapter VII. The method will be presented in detail in Section III.2.2.

II.3.1.2 Unsupervised hybrid method

These are approaches which use a mixture of raw corpus data and knowledge from an explicit knowledge base.

Yarowsky (1992) proposed a method to disambiguate words according to their category from Roget's International Thesaurus, using statistical models of the categories in the thesaurus being inferred from raw, untagged text.

Training was carried out on a corpus of 10 million words, the electronic version of Grollier's Encyclopedia.

Statistical models for each Roget category were built by extracting context around each instance of any word in the category in the corpus. The models themselves were based on Bayes' rule. Disambiguation is carried out by examining the context by examining the context of ambiguous word in text and calculating the most likely of the possible categories by applying Bayes' formula. This method was tested on 12 polysemous words (previously disambiguated words), and achieved 92% correct disambiguation.

In (McCarthy *et al.*, 2004) they introduce a predominant sense acquisition method, which consist of two steps. In the first step, a corpus of untagged text from the target domain was used to construct a thesaurus of similar words, based on distributional similarity measures. In the second step, each target word was disambiguated using pairwise WordNet-based similarity measures, taking as pairs the target word and each of the most related words according to the thesaurus up to a certain threshold. We will get back to this method in Section II.5.3.

II.3.2 Supervised WSD techniques

Supervised methods are those which rely on hand-tagged. Màrquez et al. in (Agirre and Edmonds, 2006) give a classification of the main approaches to supervised WSD. They differentiate among *probabilistic methods* (such as Naïve Bayes, Maximum Entropy), which usually estimate a set of probabilistic parameters that express the conditional or joint probability distributions of a category and context; *methods based on the similarity of the examples* (such as the Vector Space Model or k Nearest Neighbors), which perform disambiguation according to some similarity metric; *methods based on discriminating rules* (Decision List and Decision Trees are the most representative approaches), which assign senses that example satisfy one or more rule; *methods based on rule combination*, like AdaBoost, which linearly combine many simple and not necessarily accurate classification rules; and *linear classifier and kernel based approaches*, which discriminate senses calculating a hyperplane in n-dimensional feature space (the most successful algorithm is the Support Vector Machines, among others, such as the Perceptron or Winnow approaches).

In the next chapter we will present the different methods that are widely used for supervised WSD, alone or in combination. Most of our experiments are performed using k Nearest Neighbor (k-NN) and Support Vector Machines (SVM). The Vector Space Model (VSM) have been only used in experiments reported in Chapter IV.

In general terms, in order to represent a particular context of the target word, we extract features from the example. Then, the ML methods below return a weight for each sense, and the sense with maximum weight is selected.

In terms of performance, corpus-based supervised methods obtain the best results. Their performance vary depending on the number of sense-tagged examples to train, but considering the all-words task (cf. Section II.4.2) as the most realistic scenario, state-of-the-art performance is between 60% and 70% of accuracy. The next Section will review this in detail.

II.4 WSD 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.

II.4.1 Measures and significance tests

In order to measure the goodness of WSD methods, we use the following measures: precision, recall, coverage, and F-score (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):

- $precision = C/A$
- $recall = C/N$
- $coverage = A/N$
- $F{-}score = (2{*}precision{*}recall)/(precision{+}recall) = (2{*}C)/(A{+}N)$

The Senseval scoring software has been used to obtain precision and recall measures during the dissertation[5].

When comparing the performance of two algorithms, statistical tests help us to know whether the observed differences in precision or recall are significant. We will applied two of these tests in some of our experiments:

- **Bootstrap resampling** (Noreen, 1989) is a statistical method for estimating the sampling distribution of an estimator by sampling randomly with replacement from the original sample, most often with the purpose of deriving robust estimates of standard deviation and get confidence intervals of the current system results. Based on those confidence intervals we can determine whether the performance difference of two systems is statistically significant.
- The **MannWhitney U test** (Mann and Whitney, 1947) is a non-parametric alternative to the two sample t-test which is based solely on the order in which the observations from the two samples fall. The null hypothesis in the MannWhitney test is that the two samples are drawn from a single population, and therefore that their probability distributions are equal. The alternative hypothesis is that one sample is stochastically greater. It requires the two samples to be independent, and the observations to be ordinal or continuous measurements, i.e. one can at least say, of any two observations, which is the greater.

II.4.2 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:

[5]`http://www.senseval.org/senseval3/scoring`

Lexical-sample task, all-words task and translation (a kind of lexical-sample task where the senses are defined by means of translations to another language, only 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 Section II.1.2, the WordNet 1.7 (pre-release) sense inventory was chosen for English.

In the English lexical-sample the best system (JHU) scored 64.2%[6], for 51.2% of the Lesk baseline. Table II.2 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).

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 different characteristics, such as the use rich features (syntactic relations, Named Entities, Semantic Codes, or WN Domains) and feature selection and weighting.

Regarding the English all-words task, the results are shown in table II.3. 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 (CNTS-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.

Next, we will describe the top-2 from the list in the following description of Senseval-2 systems.

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

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

Rank	**System**	**Precision**	**Recall**	**Coverage**
1	JHU (R)	64.2	64.2	100.0
2	SMUls	63.8	63.8	100.0
3	KUNLP	62.9	62.9	100.0
4	Stanford - CS224N	61.7	61.7	100.0
5	Sinequa-LIA - SCT	61.3	61.3	100.0
6	TALP	59.4	59.4	100.0
7	Duluth 3	57.1	57.1	100.0
8	UMD - SST	56.8	56.8	99.9
9	BCU - ehu-dlist-all	57.3	56.4	98.3
10	Duluth 5	55.4	55.4	100.0
11	Duluth C	55.0	55.0	100.0
12	Duluth 4	54.2	54.2	100.0
13	Duluth 2	53.9	53.9	100.0
14	Duluth 1	53.4	53.4	100.0
15	Duluth A	52.3	52.3	100.0
16	Duluth B	50.8	50.8	9.9
17	UNED - LS-T	49.8	49.8	99.9
18	Alicante	42.1	41.1	97.7
19	IRST	66.5	24.9	37.4
20	BCU - ehu-dlist-best	82.9	23.3	28.0

Table II.2: 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.

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

Table II.3: 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.

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 III.2.1.2), Bayesian models (word-based and lemma-based), and Decision Lists. 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 architecture has two main components: Instance Based Learning (IBL)[7], 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 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).

For IBL, TiMBL (Daelemans *et al.*, 2007) is used with information-gain feature weighting. The novelty of this work is that they perform feature selection per each word, using cross-validation in training data.

CNTS-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 memory-based learning (TiMBL), trained each on local and topical feature space, and the rule induction Ripper algorithm. Their method scored second in the all-words task, with 63.6% precision and recall.

II.4.3 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, 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.

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

Rank	System	Team	Precision	Recall
1	htsa3	University of Bucharest	72.9	72.9
2	IRST-Kernels	ITC-IRST	72.6	72.6
3	nusels	National University of Singapore	72.4	72.4
4	htsa4	University of Bucharest	72.4	72.4
5	BCU comb	Basque Country University	72.3	72.3
6	htsa1	University of Bucharest	72.2	72.2
7	rlsc-comb	University of Bucharest	72.2	72.2
8	htsa2	University of Bucharest	72.1	72.1
9	BCU english	Basque Country University	72.0	72.0
10	rlsc-lin	University of Bucharest	71.8	71.8
11	HLTC HKUST all	HKUST	71.4	71.4
12	TALP	U.P. Catalunya	71.3	71.3
13	MC-WSD	Brown University	71.1	71.1
14	HLTC HKUST all2	HKUST	70.9	70.9

Table II.4: 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.

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.1.2.2. 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. The performance of the top-14 supervised systems is given in table II.2[8]. 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 small differences in performance for this task, where the top-9 systems are less than a point below. 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 III.2.1.4), which have been used by most of the top systems.The top-ranked systems relied on combinations

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

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

Table II.5: Top-10 systems in the Senseval-3 all-words task. For each system, the precision/recall figures are given.

to integrate different knowledge source (using kernels or voting systems) and the use of complex features. We will describe the top two systems (Htsa3 and ITC-IRST) in detail 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.3. The suffix (-S) in the name of the system indicates "supervised", and the suffix (-U) indicates unsupervised.

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. We will describe the two best-systems (GAMBL-AW and SenseLearner) below.

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 a linear kernel 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. They normalized its weight-values by dividing them with the empiric frequency of the senses in training data, which gives a higher "a posteriori" probability to frequent senses. A new parameter α is introduced to perform the normalization step smoothly.

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 III.2.1.4), 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*. The former is implemented by splitting further the kernel in *collocation kernel* (based on lemma sequences) and *PoS Kernel* (bases on PoS sequences). The latter is also the addition of another two: *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. Note that this last kernel is related to our approach in this dissertation (cf. Section III.3).

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

This system was the winner of the all-words task. 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.

GAMBL applies a word-expert approach using TiMBL and optimization of features and parameters. They apply a cascaded architecture, where a keyword-based classifier assigns a sense to the new example and it is used as feature for the second local feature based 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.

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.

II.4.4 WSD systems in SemEval-2007

The first edition of SemEval-2007[9] took place in Prague, in June 23-24, in conjunction with the ACL conference. Eighteen tasks were organized, and over 100 teams and 125 unique systems participate on them. Among the

[9]Taking into account the previous Senseval competition, actually, we can consider the fourth edition of Senseval-like competition

tasks, the typical WSD tasks, including lexical-sample and All-Words, were presented.

Task-17 (English lexical sample, SRL and all-words) focuses on both challenges, WSD and SRL, using annotated English text taken from the Wall Street Journal and the Brown Corpus. It includes three subtasks: i) the traditional All-Words task comprising fine-grained word sense disambiguation using a 3,500 word section of the Wall Street Journal, annotated with WordNet 2.1 sense tags, ii) a Lexical Sample task for coarse-grained word sense disambiguation on a selected set of lexemes, and iii) Semantic Role Labeling, using two different types of arguments, on the same subset of lexemes. We will focus on the two first sub-tasks, both related to WSD.

Rank	**Participant**	**System**	**Classifier**	**F**
1	Cai Junfu	NUS-ML	SVM	88.7±1.2
2	Oier Lopez de Lacalle	UBC-ALM	SVD+kNN	86.9±1.2
3	Zheng-Yu Niu	I2R	Supervised	86.4±1.2
4	Lucia Specia	USP-IBM-2	SVM	85.7±1.2
5	Lucia Specia	USP-IBM-1	ILP	85.1±1.2
5	Deniz Yuret	KU	Semi-supervised	85.1±1.2
6	Saarikoski	OE	naive Bayes, SVM	83.8±1.2
7	Univ. of Technology Brno	VUTBR	naive Bayes	80.3±1.2
8	Ana Zelaia	UBC-ZAS	SVD+kNN	79.9±1.2
9	Carlo Strapparava	ITC-irst	SVM	79.6±1.2
10	Most Freq. Sense in training	Baseline	N/A	78.0±1.2
11	Toby Hawker	USYD	SVM	74.3±1.2
12	Siddharth Patwardhan	UMND1	Unsupervised	53.8±1.2
13	Saif Mohammad	Tor	Unsupervised	52.1±1.2
-	Toby Hawker	USYD*	SVM	89.1±1.2
-	Carlo Strapparava	ITC*	SVM	89.1±1.2

Table II.6: System Performance for the OntoNotes Lexical Sample task. System marked with an * were post-competiton bug-fix submissions

The main important changes on lexical-sample was the use of different sense-inventory. In order to guarantee a high annotator agreement, they use OntoNotes data (Hovy *et al.*, 2006), including word senses, at high inter-annotator agreement (ITA) of over 90%. All the data for this task comes from the 1M word WSJ Treebank. They selected a total of 100 lemmas (65 verbs and 35 nouns) considering the degree of polysemy and total instances that were annotated. All the F-scores[10] in Table II.6 and Table II.7 are

[10]For a system that attempts all the words, both Precision and Recall are the same.

accompanied by a 95% confidence interval calculated using the bootstrap resampling procedure.

A total of 13 systems were evaluated on the Lexical Sample task. Table II.6 shows the Precision/Recall for all these systems. As expected, the lexical sample task using coarse grained senses provides consistently higher performance than previous more fine-grained Lexical Sample Tasks. Note that the best system performance is now closely approaching the ITA for this data of over 90%.

Regarding the system ranking, top scoring system are those based in kernel based methods (e.g. SVM) and memory-based k-NN approaches (submitted by us and detailed in Chapter IV). The results show that approaches like SVMs are the best option when enough training instance are available. Note that 5 systems out of 13 were SVM systems. The second best system was developed during this dissertation, and will be explained in Section IV.5. The best system and the third best systems (NUS-ML and I2R) will be detailed below.

Rank	**Participant**	**System ID**	**Classifier**	**F**
1	Stephen Tratz	PNNL	MaxEnt	59.1±4.5
2	Hwee Tou Ng	NUS-PT	SVM	58.7±4.5
3	Rada Mihalcea	UNT-Yahoo	Memory-based	58.3±4.5
4	Cai Junfu	NUS-ML	naive Bayes	57.6±4.5
5	Oier Lopez de Lacalle	UBC-ALM	kNN	54.4±4.5
6	David Martinez	UBC-UMB-2	kNN	54.0±4.5
7	Jonathan Chang	PU-BCD	Exponential Model	53.9±4.5
8	Radu ION	RACAI	Unsupervised	52.7±4.5
9	Most Frequent WordNet Sense	Baseline	N/A	51.4±4.5
10	Davide Buscaldi	UPV-WSD	Unsupervised	46.9±4.5
11	Sudip Kumar Naskar	JU-SKNSB	Unsupervised	40.2±4.5
12	David Martinez	UBC-UMB-1	Unsupervised	39.9±4.5
14	Rafael Berlanga	tkb-uo	Unsupervised	32.5±4.5
15	Jordan Boyd-Graber	PUTOP	Unsupervised	13.2±4.5

Table II.7: System Performance for the all-words task

Regarding the All-Words task, the data was selected from the WSJ corpus that had been Treebanked and PropBanked. As said before, WordNet 2.1 was used as sense-inventory to annotated a total of 465 lemmas from about 3500 words of text. For this dataset, they got a ITA of 72% on verbs and 86% for nouns.

A total of 14 systems were evaluated on the all-words task. These results are shown in Table II.7. The baseline performance using the most frequent WordNet sense for the lemmas is 51.4. The top-performing system was a supervised system that used a Maximum Entropy classifier, and got a Precision/Recall of 59.1% – about 8 points higher than the baseline. Below, we detail the two top ranked systems (PNNL and NUS-PT).

Compared to previous exercises, the system performance has ranged from 59% (MFS) to 65.2% (Senseval3, (Decadt *et al.*, 2004)) to 69% (Seneval2, (Mihalcea and Moldovan, 2001)). In this last edition there were proportionally more verbs and fewer nouns than previous all-words English tasks, which may account for the lower scores.

Regarding unsupervised systems, it is worth mentioning the RACAI system, which has ranked 8 and outperformed the MFS baseline. However, it is important to note that they use the sense order coded in WordNet, which is derived from Semcor sense counts and thus the MFS baseline. They obtain 52.7% F-score, which degrades to 44.5% of F-score when no sense order is used. The rest of the unsupervised systems are well below the MFS result.

NUS-ML (Cai *et al.*, 2007)

This system was the winner of the lexical-sample task with 88.7% of F-score. They relied on the data provided by the organization to train their WSD system. The submitted system is based on a rich set of feature types, which include PoS of the surrounding words to the target, local-collocations, syntactic patterns, and bag-of-words (BoW) features. In addition, they construct a topic feature targeted to capture the global context information using the latent dirichlet allocation (LDA) algorithm with unlabeled corpus. LDA clusters BoW features and thus helps relieve the scarcity problem. They used unlabeled data to cluster words into a number of pre-fixed topics. Finally, a modified na ive Bayes (NB) classifier is constructed to incorporate all the features.

I2R (Niu *et al.*, 2007)

They ranked third in the lexical-sample task of SemEval-2007, with 86.4% F-score. In order to construct the classifiers they only relied in provided data. The knowledge sources include the following features to capture the contextual information: PoS of neighboring words with the position information, the BoW features from all the contextual sentences, and local-collocations. They use a label propagation algorithm, where the label information of any vertex in a graph is propagated to nearby vertices through weighted edges until a global stable stage is achieved. They use unlabeled data in order to improve the connection within the graph.

PNNL (Tratz *et al.*, 2007)

PNNL was the best system in the all-words task, obtaining 59.1% F-score. They recollected extra training data to improve the WSD classifier: they use SemCor, OMWE (Mihalcea and Chklovski, 2003) and the example sentences in WordNet (version 2.1). A large number of features are used as a knowledge source, which consists of contextual information (3 tokens on each side of the target word), syntactic information (grammatical and morphosyntactic information) and semantic information (Named entities and hyperonyms, e.g: Joe Smith is used as PERSON). Over this large set of features an optimization is performed by selecting the k best features according to the Information Gain measure. The best features are used to train a Maximum Entropy classifier.

NUS-PT (Chan *et al.*, 2007b)

With 58.7% of F-score they got the second position in the all-words task. This team also tries to augment the training data using other resources beside SemCor corpus. English-Chinese parallel corpora and the DSO corpus are also used as training corpus. Roughly speaking, a maximum of 1000 examples were collected for each noun type word, and a maximum of 500 for verbs. The training set was built adding DSO and parallel corpora examples to SemCor examples following the sense distribution in SemCor. The SVM classifier (cf. Section III.2.1.4) was built using local-collocations, PoS and BoW as features. They omit syntactic relations for efficiency reasons.

II.5 Domain-specific WSD

First, we will briefly mention current lexical-sample datasets for domain-specific WSD. Section II.5.2 presents some possible settings for domain adaptation and Section II.5.3 reviews the state-of-the art in domain-specific WSD.

II.5.1 Domain-related datasets

We will briefly present the three datasets which have been used for domain-related studies in WSD, all of which are lexical-sample corpora.

The most commonly used dataset is the previously mentioned Defense Science Organization (DSO) corpus, which was built on purpose to study the cross domain adaptation. Another publicly available dataset is the one presented in (Koeling *et al.*, 2005), the Domain-Specific Sussex corpus. This dataset has played a central role in domain adaptation and domain-specific WSD experiments for this dissertation. Finally, a dataset for biomedicine, the National Library of Medicine WSD Test Collection, has to be mentioned, which is a useful dataset to study very specific domains. These three datasets have been described in Section II.1.2.2, and have been used in the experiments to be reported in Section II.5.3.

The biomedicine corpus tackles scholarly text of a very specific domain, and while interesting, it is not possible to apply domain adaptation techniques, as the senses used are not related to those found in generic resources like WordNet and Semcor. The DSO corpus includes texts from the Brown balanced corpus and the WSJ corpus, allowing for domain adaptation experiments. The WSJ includes texts from Finance in the widest expression, but also news of general interest which have no strict relation to the finance domain. The Sussex corpus explicitly differentiates between texts from the Finance and Sports domain, and texts from the balanced BNC, allowing for more controlled domain adaptation experiments in different settings for domain adaptation (as defined below).

Although these three corpora are useful for WSD research, they only include examples for a handful of words, and it is thus difficult to infer which would be the performance of a WSD system on full texts. The corpus of Koeling et al., for instance, only includes words which where salient for the target domains, but the behavior of WSD systems on other words can't be explored.

II.5.2 Possible settings for domain adaptation

When performing supervised WSD on specific domains the first setting is to train on a general domain data set and to test on the specific domain (**source setting**). If performance would be optimal, this would be the ideal solution, as it would show that a generalistic WSD system (trained on a generic corpus) is robust enough to tackle texts from new domains, and domain adaptation would not be necessary.

The second setting (**target setting**) would be to train the WSD systems using examples from the target domain. If this would be the optimal setting, it would show that there is no cost-effective method for domain adaptation. WSD systems would need fresh examples every time they were deployed in new domains, and examples from general domains could be discarded. In domain adaptation experiments we will give some results related to target setting as reference to other setting.

In the third setting, the WSD system is trained with examples coming from both the general domain and the specific domain. Good results in this setting would show that **supervised domain adaptation** is working, and that generalistic WSD systems can be supplemented with hand-tagged examples from the target domain.

There is an additional setting, where a generalistic WSD system is supplemented with untagged examples from the domain. Good results in this setting would show that **semi-supervised domain adaptation** works, and that generalistic WSD systems can be supplemented with untagged examples from the target domain.

For this dissertation we focused on the last two settings: **semi-supervised domain adaptation** and **supervised domain adaptation**. The semi-supervised setting will be described in Chapter V, and the supervised one will be described in Chapter VI.

The source setting and the target setting can be seen as baseline and upperbound, respectively, for the semi-supervised setting. In the case of the supervised adaptation setting, the target setting can be considered as a high baseline, as very few systems have managed to improve over it.

II.5.3 State-of-the-art in WSD for specific domain

Initial work on domain adaptation for WSD systems showed that WSD systems were not able to obtain better results on the source or adaptation settings compared to the target settings (Escudero *et al.*, 2000; Martínez and Agirre, 2000), showing that if a generic WSD system (i.e. based on hand-annotated examples from a generic corpus) would need to adapt it to a specific domain, it would be better off throwing away the generic examples and hand-tagging domain examples directly.

Escudero *et al.* (2000) tested the supervised adaptation scenario on the DSO corpus, which had examples from the Brown Corpus and Wall Street Journal corpus. They found that the source corpus did not help when tagging the target corpus, showing that tagged corpora from each domain would suffice, and concluding that hand tagging a large general corpus would not guarantee robust broad-coverage WSD. Martínez and Agirre (2000) also used the DSO corpus in the supervised scenario to show that training on a subset of the source corpora that is topically related to the target corpus does allow for some domain adaptation.

Similarly to these works, this dissertation also shows that the performance of a generic WSD system decreases when such system is trained and tested on different domains, and will propose a method to overcome this.

Better results have been obtained using purpose-built adaptation methods. Chan and Ng (2007) performed supervised domain adaptation on a manually selected subset of 21 nouns from the DSO corpus. They used active learning, count-merging, and predominant sense estimation in order to save target annotation effort. They showed that adding just 30% of the target data to the source examples the same precision as the full combination of target and source data could be achieved. They also showed that using the source corpus allowed to significantly improve results when only 10%-30% of the target corpus was used for training. Unfortunately, no data was given about the target corpus results, thus failing to show that domain-adaptation succeeded. In follow-up work (Zhong *et al.*, 2008), the feature augmentation approach was combined with active learning and tested on the OntoNotes corpus, on a large domain-adaptation experiment. They reduced significantly the effort of hand-tagging, and obtained domain-adaptation for smaller fractions of the source and target corpus. Similarly to these works we show that we can save annotation effort on the target corpus, but, in contrast, we do get domain adaptation when using the full dataset. In a way, our approach is complementary, and we could also apply active learning to further reduce the number of target examples to be tagged.

The scarce positive results in WSD contrasts with domain adaptation experiments in other NLP areas. In the supervised setting, a paper by Daumé III (2007) shows that a simple feature augmentation method for SVM is able to effectively use both labeled target and source data to provide the best domain-adaptation results in a number of NLP tasks. His method improves or equals over previously explored more sophisticated methods (Daumé III and Marcu, 2006; Chelba and Acero, 2004). The feature augmentation consists in making three version of the original features: a general, a source-specific and a target-specific versions. That way the augmented source contains the general and source-specific version and the augmented target data general and specific versions. The idea behind this is that target domain data has twice the influence as the source when making predictions about test target data.We reimplemented this method in order to compare it to our methods.

There are no many works on semi-supervised domain adaptation in NLP. Blitzer *et al.* (2006) used Structural Correspondence Learning and unlabeled data to adapt a Part-of-Speech tagger. They carefully select so-called pivot features to learn linear predictors, perform SVD on the weights learned by the predictor, and thus learn correspondences among features in both source and target domains. Our technique also uses SVD, but we directly apply it to all features, and thus avoid the need to define pivot features. In preliminary work we unsuccessfully tried to carry along the idea of pivot features to WSD. This dissertation will show show that methods based on SVD with unlabeled data and combination of distinct feature space produce positive semi-supervised domain adaptation results for WSD.

Regarding unsupervised methods, the unsupervised predominant sense acquisition method was introduced first in (McCarthy *et al.*, 2004). (McCarthy *et al.*, 2004) does not report any domain adaptation experiments, but in follow-up work,(Koeling *et al.*, 2005; McCarthy *et al.*, 2007) report results on domain specific corpora. The predominant sense acquisition method consisted basically on two steps. In the first step, a corpus of untagged text from the target domain was used to construct a thesaurus of similar words. In the second step, each target word was disambiguated using pairwise WordNet-based similarity measures, taking as pairs the target word and each of the most related words according to the thesaurus up to a certain threshold. This method aims to obtain, for each target word, the sense which is the most predominant for the target corpus. When a general corpus is used, the most predominant sense in general is obtained (McCarthy *et al.*, 2004), and when a domain-specific corpus is used, the most predominant sense for that corpus is obtained (Koeling *et al.*, 2005). The main motivation of the authors is that the most frequent sense is a very powerful baseline, but it is one which requires hand-tagging text, while their method yields similar information automatically. The results show that they are able to obtain good results. This dissertation revisits their proposal, and confirms the good results of their approach, specially in specific domains.

Current research on applying WSD to specific domains has been evaluated on three available lexical-sample datasets (Ng and Lee, 1996; Weber *et al.*, 2001; Koeling *et al.*, 2005). This kind of datasets contains hand-labeled examples for a handful of selected target words. As the systems are evaluated on a few words, the actual performance of the systems over complete texts can not be measured. Actually, the different behavior of WSD systems when applied to lexical-sample and all-words datasets has been observed on previous Senseval and Semeval competitions (Kilgarriff, 2001; Mihalcea *et al.*, 2004; Pradhan *et al.*, 2007), where supervised systems attain results on the high 80's and beat the most frequent baseline by a large margin for lexical-sample datasets. The results on the all-words datasets were much more modest, on the low 70's, and a few points above the most frequent baseline.

CHAPTER III

Methods and approaches from this dissertation

In this chapter we will present the main elements used in the dissertation. The chapter is organized as follows. First, we will briefly introduce the learning features used in the supervised classifiers. The next section will describe the WSD approaches used in this work, both supervised and knowledge-based. The next section will be devoted to present Singular Value Decomposition (SVD). We will motivate the usefulness of SVD, some mathematical foundation will be explained, and application to WSD will be described, as well as prior work on SVD for WSD. Finally, the last section will describe the proposed classifier combinations, both k based on nearest neighbors and kernels.

III.1 Learning features

In supervised learning for classification the consists in inducing an approximation (or hypothesis) h of an unknown function f that maps from the input space X to a discrete unordered output space $Y = \{1, ..., K\}$, given a training set S, .

The training set contains m examples, $S = \{(\mathbf{x}^1, y^1), ..., (\mathbf{x}^m, y^m)\}$, where in each pair $(\mathbf{x}, y)$, $\mathbf{x}$ belongs to X and $y = f(\mathbf{x})$. $\mathbf{x}$ is typically a vector $\mathbf{x} = (x_1, ..., x_n)$, whose components, called *features*, are discrete- or real-valued and describe the relevant information or properties about the example.

The values of the output space Y associated with each training example are called *classes*. Therefore, each training example is completely described by set of feature-value pairs and class label. In our case the examples are occurrences of the target words and the classes are the senses of the target words that apply to those occurrences.

Features try to capture the information and knowledge about the context of the target words to be disambiguated. Computational requirements of learning algorithms and the availability of the information impose some limitations on the features that can be considered, thus they necessarily codify only a simplification (or generalization) of the occurrence of word sense instances (See Section II.2 for more details on knowledge representation for WSD).

Usually, a complex pre-processing step is performed to build a feature vector for each context of occurrence. This step often considers the use of windowing or sentence-splitting, a PoS tagger to establish PoS patterns around the target word, *ad hoc* routines for detecting multi-words or capturing *n-grams*, or parsing tools for detecting syntactic dependencies between lexical units.

The state-of-the-art on WSD has shown that all types of features are necessary to assure good performance of supervised ML approaches. An interesting survey on feature types and the knowledge they code is given in (Agirre and Martínez, 2001a).

The feature types used in this dissertation can be grouped in four main sets. These features are described and analyzed in (Martínez, 2004):

Local collocations: Bigrams and trigrams formed with the words around the target. These features are constituted by lemmas, word-forms, or PoS tags[1]. Other local features are those formed with the previous/posterior lemma/word-form in the context.

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.

Bag-of-words features: We extract the lemmas of the content words in the

[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 the Johns Hopkins University.

```
<instance id=1“:0@67@brown/cr/cr08@brown@en@on” docsrc=“wsj”>
<answer instance= “1:brown/cr08@brown@en@on” senseid=“1”/>
<context>
In this work, his use of non-color is startling and skillful. The sweep of space,
the delicate counterbalance of the white masses , the over-all completeness
and unity, the originality and imagination , all entitle it to be called an
authentic masterpiece. I <head> asked </head> Quasimodo recently
how he accomplished this, and he replied that he had painted his model “a
beautiful shade of red and then had her breathe on the canvas”, which was
his typical tongue-in-cheek way of chiding me for my lack of sensitivity .
</context>
</instance>
```

Figure III.1: An example from the SemEval-2007 lexical sample dataset, showing the context of occurrence for *ask*, which is the target word.

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).

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].

Domain features were only used in our SemEval lexical sample and all words dataset (for more details see Chapter IV. In the rest of the experiments we worked with the other three feature sets (local collocations, syntactic dependencies, bag-of-words). We will refer to this later in Section IV.2.2.

Figure III.1 shows an example extracted from SemEval-2007 lexical-sample task. The examples consist of a context for the target word *ask* (actually *asked*), where the size of the context consist of the previous two sentences, the sentence containing the target word, and the following two sentences. The features extracted from this example are related to the target word, as the following sample shows:

- Regarding local collocations, “*post_N_lem Quasimodo*” would indicate

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

that the target word is followed by the noun *Quasimodo*. Similarly, "*post_J_lem accomplished*" and "*post_R_lem recently*" would indicate that *accomplished* is the following adjective and *recently* is the next adverb to the target word, respectively.

- An instance for a syntactic dependencies features would be "*Object Quasimodo_NNP*" where *Quasimodo* is the object of the target verb *ask*.

- Concerning bag-of-words features, we would have one feature for each lemmas in this chunk of text: "*win_cont_lem_context breath*". "*pedersen_bigr beautiful shade*" would indicate that these salient bigram occurs in context (Pedersen, 2001).

- Domain features depend on the words in the context. In this case, the domain tags would be "*Domain_A factotum*" and "*Domain_A quality*", where "*Domain_F quality*" is the most relevant domain tag.

These four sets of feature will be referred as **original learning features**. The whole set of feature type is listed in Appendix B. Section III.3 will describe how SVD is used to produce new features, which will be referred to as as SVD **learning features**.

III.2 WSD Algorithms

In this section we will introduce the algorithms which have been used in the dissertation. These algorithms are considered to be representative of the state-of-the-art in WSD. We will first describe supervised methods, followed by a knowledge based method (Agirre and Soroa, 2009).

III.2.1 Supervised methods

Supervised methods rely on hand-tagged data in order to induce models and disambiguate test examples. To date they outperform other methods. See Section II.3.2 for a short review of methods. The we will now review the methods used in our dissertation: Most Frequent Sense heuristic (MFS), the Vector Space Model (VSM; Section III.2.1.2), k Nearest Neighbor (k-NN, Section III.2.1.3) and Support Vector Machines (SVM, Section III.2.1.4).

III.2.1.1 Most Frequent Sense baseline (MFS)

This simple baseline method is frequently applied in WSD. 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, and more complex supervised systems sometimes only beat it by a small margin.

III.2.1.2 Vector Space Model (VSM)

In the Vector Space Model 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 ($\boldsymbol{C_{s_k}}$). These centroids are compared with the vectors that represent testing examples ($\boldsymbol{f}$), by means of the cosine similarity function (formula III.1). The closest centroid assigns its sense to the testing example.

$$weight(s_k) = cos(\boldsymbol{C_{s_k}}, \boldsymbol{f}) = \frac{\boldsymbol{C_{s_k}} . \boldsymbol{f}}{||\boldsymbol{C_{s_k}}|| ||\boldsymbol{f}||} \tag{III.1}$$

Leacock *et al.* (1993) compared SVM, Neural Networks and NB and drew the conclusion that the two first methods slightly surpass the last one in WSD. In Senseval-2 a combination of different methods, including SVM, obtained very good results (Yarowsky *et al.*, 2001). This method's performance is reported in Chapter IV.

III.2.1.3 Memory-Based Learning (k-NN)

The most widely used representative of this family is the k Nearest Neighbor. In this algorithm, the classification of a new example is performed by searching the set of the k most similar examples (or *nearest neighbors*) among the previously stored set of sense-tagged examples (the training part). The sense used most in those k examples is selected. In the simplest case, which has been used in this dissertation, the training step is reduced to storing all of the examples in memory (thus is being called memory-based) and the generalization is postponed until each new example is classified. This is why it is sometimes also called *Lazy Learning.*

An important issue in this method is the definition of a similarity metric for the task. We chose the cosine measure to calculated the similarities among the feature-vectors (cf. Eq. (III.1)). The combination is scheme for deciding the resulting sense among the k nearest neighbors is an issue to be aware of. In our case, the sense is selected by the maximum sum of the weighted votes of the k most similar contexts. Votes have been weighted depending on its (neighbor) position in the ordered rank, with the closest being first. Eq. (III.2) formalizes k-NN algorithm, where C_i corresponds to the sense label of the i-th closest neighbor. Although we are aware of other voting schemes and similarity metrics (Daelemans and van den Bosch, 2005), in preliminary experiments we realized that the method used here has an good performance for WSD.

$$\arg\max_{S_j} = \sum_{i=1}^{k} \begin{cases} \frac{1}{i} & \text{if } C_i = S_j \\ 0 & \text{otherwise} \end{cases} \tag{III.2}$$

k-NN is mentioned to be the best option for WSD by Ng (1997). Other authors (Daelemans *et al.*, 1999) argue that memory-based methods tend to be superior in NLP problems because they do not applied any kind of generalization and, therefore, they do not forget exceptions.

In Section III.4.1 we will describe our proposal for combining several k-NN classifiers, which has been shown to be a robust system for domain adaptation task.

III.2.1.4 Support Vector Machines (SVM)

Support Vector Machines (SVM) are based on the *Structural Risk Minimization* principle from Statistical Learning Theory (Vapnik, 1998). Basically, they construct a separating hyperplane that divide a set of positive examples from a set of negative examples. The location of this hyperplane in the space maximizes the distance between the closest positive and negatives examples. Experimental results have shown that this learning method has good properties in terms of generalization bounds for the induced classifiers. Figure III.2 shows the geometrical intuition of the maximal margin hyperplane in two-dimensional space. The linear classifier is composed by two main elements, which define the hyperplane: (1) a weight vector $\boldsymbol{w}$ (give some weight to each feature) and (2) a bias b which stands for the distance of the hyperplane to the origin. The classification rule assigns $+1$ or -1 to a new example $\boldsymbol{x}$ as follows:

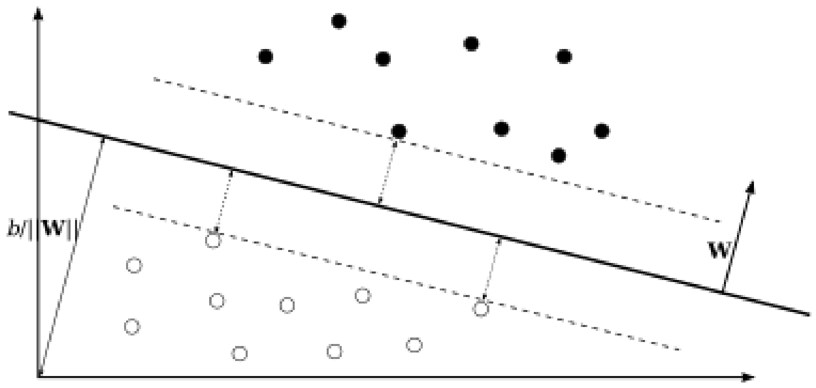

Figure III.2: Geometrical interpretation of Support Vector Machines.

$$h(\boldsymbol{x}) = \begin{cases} +1 & \text{if } \langle \boldsymbol{w} \cdot \boldsymbol{x} \rangle + b \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{III.3}$$

The examples (positive and negative) closest to the hyperplane are the *support vectors*.

Learning the maximal margin hyperplane $(\boldsymbol{w}, b)$ is stated as a convex quadratic optimization with a unique solution in its primal form: minimize $||\boldsymbol{w}||$ subject to (one of each example) $y_i[< \boldsymbol{w} \cdot \boldsymbol{w_i} \geq 1]$, indicating that all training examples are classified with a margin equal or greater than 1.

Interestingly, it is possible to obtain a dual representation (*Lagrangian formulation*) in which only training examples will be taken into consideration in the form of dot products between vectors:

$$\begin{aligned} \max \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j \langle \boldsymbol{x_i} \cdot \boldsymbol{x_j} \rangle \\ \text{subject to: } \sum_{i=1}^{N} \alpha_i y_i = 0,\ \alpha \geq 0,\ \forall 1 \leq i \leq m \end{aligned} \tag{III.4}$$

From this formulation, the orthogonal vector to the hyperplane rests defined as:

$$h(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = \sum_{i=1}^{m} \alpha_i y_i \langle \boldsymbol{x_i} \cdot \boldsymbol{x} \rangle \tag{III.5}$$

Where $\alpha_i \neq 0$ for all training examples in the margin (support vectors) and $\alpha_i = 0$ for the others. The classification rule is defined as in Eq. (III.3).

Duality is an important property of linear learning machines, which has been a crucial concept in the development of SVM. Using the dual form, data appears only within dot products, and allows the use of *kernel functions* to produce more powerful classifiers such as non-linear classifiers.

Kernel functions make SVM work efficiently and implicitly in very high dimensional features spaces, where new features can be expressed as combinations of many basics features (Cristianini and Shawe-Taylor, 2000).

We will extend the explanations about kernel functions in the next section.

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). This method has been used during the whole dissertation.

III.2.1.5 Implicit mapping: the kernel trick

Kernel methods are a well known approach to solve ML problems. Given the dual form, in which the training data only will appear in the form of dot product, we can transform the feature space in a manner that non-linear generalization are possible. In other words, kernel representations offer an alternative solution by projecting the data into more informative new feature spaces to increase the computational power of linear classifiers.

The possibility to map easily comes from the characteristic of representation in a dual form of any linear classifier (including support vector machines). The main advantage of using the dual representation is that the number of tunable parameters does not depend on the number of attributes being used. This means that the hypothesis can be expressed as a linear combination of the training points, and the decision can be taken calculating the inner products of the test point and the training points. In this sense, it is clear that defining an appropriate kernel function allows one to rewrite the data in the new representation. Depending on the problem, choosing a correct kernel will correspond to implicitly choosing a correct feature space mapping, since the kernel function is defined by, $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$, where ϕ is

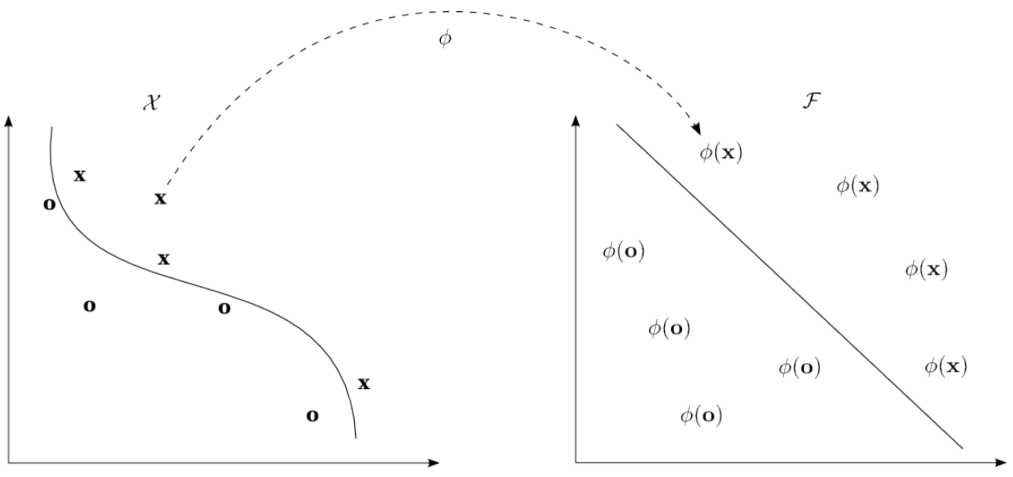

Figure III.3: A feature map can simplify the classification task

the mapping function, $\phi : \mathcal{X} \to \mathcal{F}$, and $\langle \cdot \rangle$ denotes a inner product between vectors in the feature space ($\mathcal{F}$).

Given a training set $S = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_m}\}$, the information to kernel based algorithms is contained entirely in the matrix of inner products,

$$G = K = \left(k\left(\mathbf{x_i} \cdot \mathbf{x_j}\right)\right)_{i,j=1}^{m},$$

known as the Gram or kernel matrix. The information within the matrix can be exploited, for instance, by operating on the matrix and giving the chance to combine kernels into a single kernel.

The set of hypotheses considered is the following linear function:

$$h\left(\mathbf{x}\right) = \langle \mathbf{w} \cdot \phi\left(\mathbf{x}\right) \rangle + b$$

Applying the kernel trick, this weighted vector can be expressed as a linear combination of the training points, $\mathbf{w}\left(\mathbf{x}\right) = \sum_{i=1}^{m} \alpha_i \phi\left(\mathbf{x_i}\right)$, and h can be expressed in a dual representation,

$$h(\boldsymbol{x}) = \langle \boldsymbol{w} \cdot \phi(\boldsymbol{x}) \rangle + b = \sum_{i=1}^{m} \alpha_i y_i k(\boldsymbol{x_i}, \boldsymbol{x})$$

Given an explicit feature map ϕ we can use $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$ to compute the corresponding kernel. The next section will describe the designed kernels and their feature mapping. All in all, in order to learn different relations with a linear machine, we need to use the set of non-linear features and to write the data in the new representation. We can see this as a way to build a new linear (which can work as a non-linear) classifiers in two step:

1. Apply any kind of mapping to the original input space, where $\phi : \mathcal{X} \rightarrow \mathcal{F}$ is a mapping function from input space $\mathcal{X}$ into feature space $\mathcal{F}$.

2. Train any linear classifier as SVM.

Figure III.3 illustrates an example of feature mapping from two a dimensional input space to a two dimensional feature space, where the data cannot be separated by a linear function in the input space, but can be separated in the mapped space.

Kernels give the chance to map (via non-linear functions) the features onto higher dimensional space where linear separation could become an easier work. Although high dimensional spaces can be more informative, they usually are computationally expensive, and sometimes might be preferable to apply some kind of dimensionality reduction. In this dissertation, we apply dimensionality reduction, moving from n dimensions to d dimensions, such as:

$$\mathbf{x} = (x_1, ..., x_n) \longmapsto \boldsymbol{\phi}(\mathbf{x}) = (\phi_1(x), ..., \phi_d(x)), d < n,$$

where the similarities between points computed in a reliable manner, and both the computational and the generalization performance improve.

III.2.1.6 Feature augmentation method

In order to asses our contribution we decided to re-implement the feature augmentation method introduced by Daumé III (2007). The main idea is to give more importance to those examples that are coming from the same domain, compared to those coming from a different domain. In this case, if we are testing in the target domain, we would prefer to give more discriminative power to the instances from training which are also drawn in the target domain. This is related to (Escudero *et al.*, 2000), where they found that source corpus does not help when tagging the target corpus.

Essentially, all we are going to do is to make three versions of the original feature vector: a general version, a source-specific version and a target-version. The augmented source data will contain only general and source-specific versions. The augmented target data contains general and target-specific versions. Thus, we will define our augmented space by $\mathcal{X}' = \mathbb{R}^{3F}$. Then, define mappings $\phi_s, \phi_t : \mathcal{X} \to \mathcal{X}'$ for mapping the source and target data respectively:

$$\phi_s(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{x}, \boldsymbol{0} \rangle, \, \phi_t(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{0}, \boldsymbol{x} \rangle \tag{III.6}$$

Suppose that the data points x are drawn from a reproducing kernel Hilbert space $\mathcal{X}$ with kernel $K : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$, K positive semi-definite. Then, K can be written as the dot product of two vectors: $K(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{x}') \rangle$. Define ϕ_s and ϕ_t in terms of ϕ, as:

$$\phi_s(\boldsymbol{x}) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}), \boldsymbol{0} \rangle, \, \phi_t(\boldsymbol{x}) = \langle \phi(\boldsymbol{x}), \boldsymbol{0}, \phi(\boldsymbol{x}) \rangle \tag{III.7}$$

Now, we denote an expanded Kernel by $K'(\boldsymbol{x}, \boldsymbol{x}')$. When the domain is the same, we get: $K'(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{x}') \rangle + \langle \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{x}') \rangle = 2K(\boldsymbol{x}, \boldsymbol{x}')$. When the domains differ, we get: $K'(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{x}') \rangle = K(\boldsymbol{x}, \boldsymbol{x}')$. Considering the kernel as a measure of similarity – data points from the same domain are *a priori* twice similar than those from different domains.

III.2.2 Knowledge based method

Among a number of graph based methods for knowledge based WSD (Sinha and Mihalcea, 2007; Navigli and Lapata, 2007; Mihalcea, 2005; Agirre and Soroa, 2008) we focused on the **Personalized PageRank** algorithm introduced by Agirre and Soroa (2009), as it has been shown to produce superior results. We have utilized this method in domain-specific WSD experiments reported in Chapter VII.

Personalized PageRank is based on the PageRank algorithm. The PageRank algorithm (Brin and Page, 1998) is a method for ranking the vertices on a graph according to their relative structural importance. The main idea of PageRank is that whenever a link from v_i to v_j exists on a graph, a vote from node i to node j is produced, and hence the rank of node j increases. Besides, the strength of the vote from i to j also depends on the rank of node i: the more important node i is, the more strength its votes will have. Alternatively, PageRank can also be viewed as the result of a random walk process, where the final rank of node i represents the probability of a random walk over the graph ending on node i, at a sufficiently large time.

Let G be a graph with N vertices $v_1, \ldots, v_N$ and d_i be the out degree of node i; let M be a $N \times N$ transition probability matrix, where $M_{ji} = \frac{1}{d_i}$ if a link from i to j exists, and zero otherwise. Then, the calculation of the *PageRank vector* **PR** over G is equivalent to solving Equation (III.8).

$$\mathbf{PR} = cM\mathbf{PR} + (1-c)\mathbf{v} \tag{III.8}$$

In the equation, $\mathbf{v}$ is a $N \times 1$ vector whose elements are $\frac{1}{N}$ and c is the so called *damping factor*, a scalar value between 0 and 1. The first term of the sum on the equation models the voting scheme described in the beginning of the section. The second term represents, loosely speaking, the probability of a surfer randomly jumping to any node. The damping factor c models the way in which these two terms are combined at each step.

The second term on Eq. (III.8) can also be seen as a smoothing factor that makes any graph fulfill the property of being aperiodic and irreducible, and thus guarantees that PageRank calculation converges to a unique stationary distribution.

In the traditional PageRank formulation the vector $\mathbf{v}$ is a stochastic normalized vector whose element values are all $\frac{1}{N}$, thus assigning equal probabilities to all nodes in the graph in case of random jumps. However, as pointed out by Haveliwala (2002), the vector $\mathbf{v}$ can be non-uniform and assign stronger probabilities to certain kinds of nodes, effectively biasing the resulting PageRank vector to prefer these nodes. Such a calculation is often called a *Personalized PageRank*. For example, if we concentrate all the probability mass on a unique node i, all random jumps on the walk will return to i and thus its rank will be high; moreover, the high rank of i will make all the nodes in its vicinity to also receive a high rank. Thus, the importance of node i given by the initial distribution of $\mathbf{v}$ spreads along the graph on successive iterations of the algorithm.

PageRank is actually calculated by applying an iterative algorithm which computes Eq. (III.8) successively until convergence below a given threshold is achieved, or, more typically, until a fixed number of iterations are executed. Following usual practice, we used a damping value of 0.85 and finish the calculations after 30 iterations. We did not optimize these parameters (further details in Chapter VII).

III.3 Singular Value Decomposition

As presented in the introductory section, SVD is a technique to reduce the dimensions a matrix, composed of vector representations of a problem. It has been widely used in Text Categorization or Information Retrieval, being the basic tool of Latent Semantic Analysis. SVD reduces the dimensionality of the feature vectors, finding correlations (both first order and higher order) between features and, as we will show in this dissertation, representing the data in a manner that helps to deal with data sparseness and domain shift problem. We will review briefly SVD as we apply it to WSD.

This section will introduce the main technique as used in this dissertation, and we will refer repeatedly to this technique . In Section I.6 we tried to motivate the use of SVD in terms of the main difficulties found by supervised classifiers in a WSD task, summarized as follows:

- **Data sparsity**: Most of the events occur rarely, even when large quantities of data are available. Thus, the high dimensional features obtained from training data are composed of mainly zeros. SVD implicitly finds correlations among features and documents, identifying synonymous or closely related words and other relations between features. We show that these relations can be exploited to alleviate the sparsity in WSD.

- **Domain shift**: Different domains involve different predominant senses, some words tend to occur in fewer senses in the specific domains, the context of the senses might change, new senses and terms might be involved. We show that SVD takes advantage of higher order correlations, and help alleviate the gap between domains.

- **Data redundancy**: SVD puts features which work similarly onto the same reduced dimension, and thus help help against the *curse of dimensionality* and can be seen as a method for feature selection.

In the next subsection we will introduce a toy example that shows the above problems. Section III.3.2 will describe some mathematical foundations regarding SVD. Finally, we will show how we applied this method to WSD.

III.3.1 A toy-example

Let $C = \{d_1, d_2, ..., d_n\}$ be a corpus, where d_i is a document from the corpus. Let $W = \{t_1, t_2, ..., t_m\}$ be the terms appeared in C, let $M \in \mathbf{R}^{m \times n}$ be a term-by-document matrix representing C, where $m_{ij} \in M$ is the frequency of term t_i in instance d_j. Note that m_{ij} can be appropriately weighted if desired. Equation (III.9) shows the example matrix, which we will use through this section. This example matrix have been taken from (Manning and Schütze, 1999).

$$M = \left(\begin{array}{l|cccccc} & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\ \hline \textit{cosmonaut} & 1 & 0 & 1 & 0 & 0 & 0 \\ \textit{astronaut} & 0 & 1 & 0 & 0 & 0 & 0 \\ \textit{moon} & 1 & 1 & 0 & 0 & 0 & 0 \\ \textit{car} & 1 & 0 & 0 & 1 & 1 & 0 \\ \textit{truck} & 0 & 0 & 0 & 1 & 0 & 1 \end{array}\right) \tag{III.9}$$

The matrix in Eq. (III.9) illustrates how data sparsity makes the overlap to be zero, both for related terms and related documents. For instance, *cosmonaut* and *astronaut*, being synonyms, do not share any document. If we used the matrix to compute similarity between terms, e.g. using the cosine between their document vectors (row vectors), we will conclude that *cosmonaut* and *astronaut* are completely dissimilar, since the cosine would yield 0.00.

In a likely manner, the similarity among documents will fail too. For example, document d_2 and document d_3 have no word in common, and thus the resulting similarity will be 0.00 as well.

The next section will review the basics notions for singular value decomposition and its low-rank approximation, and show how it can find correlations between words and documents.

III.3.2 SVD: Mathematical foundations

In this Section we will explain SVD, then main properties of low-rank approximation and how we can take advantage of them, and its application to WSD.

III.3.2.1 Matrix and SVD decomposition

Given a $m \times n$ term-by-document matrix M, let U be a $m \times m$ matrix whose columns are orthogonal eigenvectors of M^TM, and V be the $m \times n$ matrix whose columns are orthogonal eigenvectors of MM^T. Following theorem III.3.1 below, SVD decomposes a matrix M as a product of three matrices.

Theorem III.3.1 *Let be k the rank of $M \in \mathbf{R}^{m \times n}$ matrix. Then, we obtain a* singular value decomposition *of the form*

$$M = U\Sigma V^T = \sum_{i=1}^{k=min\{m,n\}} \sigma_i u_i vi^T \qquad \text{(III.10)}$$

where

- $U = [u_1, ..., u_m] \in \mathbf{R}^{m \times m}$*:* u_i *columns are orthonormal and called* left singular *vectors of* M*.*
- $V = [v_1, ..., v_m] \in \mathbf{R}^{m \times n}$*:* u_i *columns are orthonormal and called* right singular *vectors of* M
- Σ *is a diagonal matrix which contains* k singular values *in descending order, where* $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_k \geq ... \geq 0$ *and* $k = min\{m, n\}$*.*

Note that in WSD the number of instances (documents) is much lower than the number of features (terms), and thus $n << m$ and k is always equal to the number of instances.

$$U = \left(\begin{array}{l|rrrrr} & dim_1 & dim_2 & dim_3 & dim_4 & dim_5 \\ \hline cosmonaut & -0.44 & -0.30 & 0.57 & 0.58 & 0.25 \\ astronaut & -0.13 & -0.33 & -0.59 & 0.00 & 0.73 \\ moon & -0.48 & -0.51 & -0.37 & 0.00 & -0.61 \\ car & -0.70 & 0.35 & 0.15 & -0.58 & 0.16 \\ truck & -0.26 & 0.65 & -0.41 & 0.58 & -0.09 \end{array}\right) \qquad \text{(III.11)}$$

$$\Sigma = \begin{pmatrix} 2.16 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.59 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.28 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.39 \end{pmatrix} \qquad \text{(III.12)}$$

$$V^T = \left(\begin{array}{c|cccccc} & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\ \hline dim_1 & -0.75 & -0.28 & -0.20 & -0.45 & 0.33 & -0.12 \\ dim_2 & -0.29 & -0.53 & -0.19 & 0.63 & 0.22 & 0.41 \\ dim_3 & 0.28 & -0.75 & 0.45 & -0.20 & 0.12 & -0.33 \\ dim_4 & 0.00 & 0.00 & 0.58 & 0.00 & -0.58 & 0.58 \\ dim_5 & -0.53 & 0.29 & 0.63 & 0.19 & 0.41 & -0.22 \end{array}\right) \quad \text{(III.13)}$$

For instance, the decomposition of the matrix III.9 is given in matrices III.11 through III.13. As mentioned before this matrices have orthonormal columns, which means that columns vectors are unit length and are orthogonal to each other ($U^TU = V^TV = I$, where I is a diagonal matrix). Matrix III.12 contains the singular values of M. Note that it is a diagonal matrix, where the singular values are stored in descending order. All singular values are semi-positive and the i^{th} singular values indicates the amount of variation among the i^{th} axis.

As intuitive explanation of SVD, it is possible to view SVD as a process where the axes are rotated in the n-dimensional space. The largest variation among the documents are represented along the first axis, the second largest variation along the second dimension and so forth until the last singular value. The matrices U and V represent terms and documents in a new space, so the first row of U corresponds to the first row of M and so on. Similarly, the first column of V corresponds to the first row of M. Further explanation will be given in the next section.

III.3.2.2 Low-rank approximation

SVD can be used to solve the **low-rank approximation** as we will show in this section. Given a $m \times n$ matrix M the low-rank approximation consist in finding the M_k matrix of rank at most k, where we minimize the discrepancy between M and M_k, calculated by the *Frobenious norm* of the matrix difference $||M - M_k||$. If $k << r$, where r is the rank of M, then this is referred to as a low-rank approximation.

Once we have decomposed the matrix M in singular values of form $M = U\Sigma V^T$, then we can select the first p singular values from Σ, where $p < k$ and k is the rank of M. Replacing by zeroes the rest of the singular values we obtain the $M_p = U\Sigma_p V^T$ as the rank-p approximation to M. The rank of the approximated matrix is at most k.

The following theorem introduced by Eckart and Young (1936) explains that among the matrices with rank p or lower, M_p is the one with the highest approximation to M.

Theorem III.3.2 *Given the decomposition in III.3.1, where the rank of* M *is* k $(k \leq min(m,n))$*. The following is defined:*

$$M_p = U_p \Sigma_p V_p^T = \sum_{i=1}^{p} \sigma_i u_i v_i^T \qquad p \leq k, \tag{III.14}$$

where U_p *and* V_p *are first* p *columns of* U *and* V*. The following is satisfied:*

$$\min_{A||rank(A) \leq p} ||M - A|| = ||M - M_p|| = \sigma_{p+1} \tag{III.15}$$

Due to Theorem III.3.2 we know that M_p is the best rank-k approximation to M and the incurring error is σ_{k+1}. Thus, we move from the vector space defined by M to the vector space defined by U_p and this new space is called the p dimensional reduced space or the **latent semantic space**.

As we mentioned before, selecting the first p dimensions we obtain the best rank-k matrix approximation. In other words, via SVD it is possible to represents the terms and the documents in a low-dimensional vector space. This is a key property to exploit correlations among words and documents. Words with similar co-occurrence patterns are projected onto the same direction. This way, semantically related words and documents will be measured as similar even if the words do not share documents or the documents do not share words.

$$B = \Sigma_2 V_2^T = \left(\begin{array}{c|cccccc} & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 \\ \hline dim_1 & -1.62 & -0.60 & -0.44 & -0.70 & -0.70 & -0.26 \\ dim_2 & -0.46 & -0.84 & -0.30 & 1.00 & 0.35 & 0.65 \end{array} \right) \tag{III.16}$$

The example in Equation (III.11) shows that the second dimension on matrix U (column dim_2) splits up the words in two groups. The first group contains the terms related to space exploration topic(*cosmonaut, astronaut, moon*) with negative values in this column, whereas the second group contains automobile-related terms (*car, truck*), with positive values.

Equation (III.16) shows the representation of the documents in two dimensions after rescaling to the first 2 singular values (with the diagonal

	d_1	d_2	d_3	d_4	d_5	d_6
d_1	1.00					
d_2	0.78	1.00				
d_3	0.40	0.88	1.00			
d_4	0.47	-0.18	-0.62	1.00		
d_5	0.74	0.16	-0.32	0.94	1.00	
d_6	0.10	-0.54	-0.87	0.93	0.74	1.00

Table III.1: The matrix of document correlation calculated with the cosine similarity. The documents are represented in 2 dimension (see Eq. (III.16)) and rescaled with the singular values: $B^TB = V\Sigma_2\Sigma_2V^T$

values 2.16 and 1.59 from Σ) in this way: Σ_2V^T. As the term-similarity has changed, it also affects document similarity. Table III.1 shows the matrix of document similarity when using the 2 dimension space. As one could expect from the contents of the documents, we can see two groups of highly similar documents, with d_1 and d_2 (0.88) and d_4, d_5 and d_6 (0.94, 0.93, 0,74) showing high similarity among them. Note that d_2 and d_3 are now highly similar, even if they don't share words any words and the similarity in the original space is 0.00.

This toy-example shows that SVD is able to bring together terms with similar co-occurrence patterns, and also makes it possible to improve the quality of any pattern recognition system based on distributional similarity. The next section will show how to extend this idea to WSD.

III.3.3 Application to WSD

Our approach is inspired by Latent Semantic Indexing (LSI), as introduced by Deerwester *et al.* (1990). This method projects unseen vectors onto the low-dimensional space collapsing terms with similar correlation patterns, such as synonyms, onto the same dimensions, incrementing the similarity among their respective vectors.

In the following sections we will detail the different approaches to build the matrices and then apply the decomposition. In general terms, we build two types of matrices:

- A unique **term-by-document** matrix for every target word to be disambiguated. We will refer to this as the **single matrix for all words** matrix. This approach is very similar to LSI and to that used in (Gliozzo *et al.*, 2005).

- A **feature-by-instances** matrix for each target word, where in the columns we use the context of each instance of the target word, and in the rows we use all features as defined in Section III.1. We will refer to these as **one matrix per target word** matrices.

After we build the matrices, we proceed to decompose it into three matrices ($M = U\Sigma V$) and then construct the reduced dimensionality space selecting the first p singular values. Thus we reduce the current space to p dimensions, and can thus project hand-tagged training and test instanced into the reduced space. The Equation (III.17) shows how to make this projection, where $\vec{t^T}$ is the transpose of the vector of features or terms corresponding to an occurrence of the target word. Thus we can easily project any vector that did not occur in the initial matrix M (typically test instances that we need to classify) into the low-dimensional space.

$$\vec{t_p} = \vec{t^T} U_p \Sigma_p^{-1} \tag{III.17}$$

Note that we can map each row (features or terms) and columns (instances or documents) to the p dimensional space, because this space is defined by the p principal eigenvectors of MM^T and M^TM.

Once we project all training and testing instances into the reduced space, we can apply any ML algorithm as usual, using the values of the reduced dimensions as new features.

In the following sections we will describe the details on matrix construction and what we would expect from them: Section III.3.3.1 describes how to use the term-by-document matrix for WSD task, Section III.3.3.1 focuses on the feature-by-instance matrix, and Section III.3.3.3 shows how to use unlabeled data in order to obtain more reliable correlations when applying SVD.

III.3.3.1 Single matrix for all words

When building a **single matrix for all words** (SMA), we define a unique matrix for all target words. The idea is to use unlabeled data obtained from any corpus to find correlations among terms occurring in the documents. Those terms are actually the values of bag-of-word features in train and test

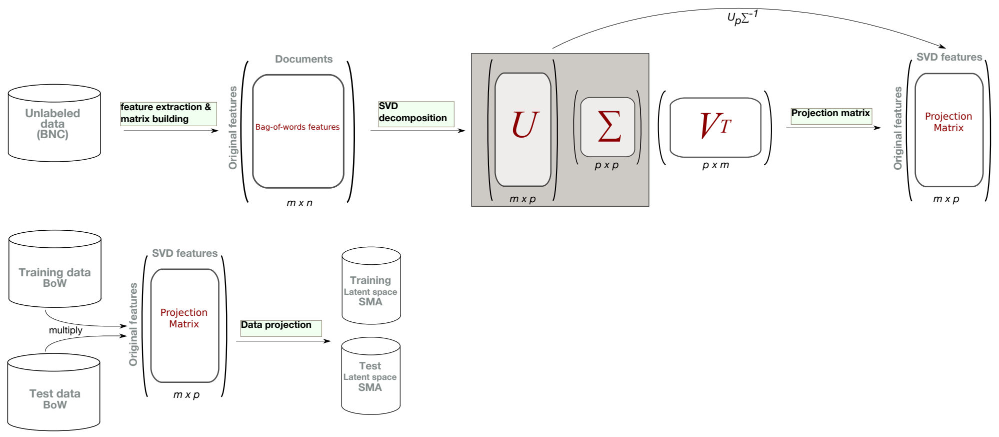

Figure III.4: Summary of the whole process of extracting the SVD features from SMA matrix: SMA features.

instances, so we are actually finding correlations between those features, and thus overcome the low overlap of features caused by data-sparseness and domain shifts.

In order to build the SMA matrix and extract the features to be used in WSD, the method comprises the following steps:

i. Extract terms from unlabeled corpora. Previously, documents are tokenized and lemmatized.

ii. Build the term-by-document matrix. Optionally, it is a good idea to apply a weighting scheme for term frequencies such as *tf-idf* or *log-entropy*.

iii. Decompose the matrix with SVD (see Eq. (III.10)) and obtain the projection matrix $U_p\Sigma_p^{-1}$ after the new dimensionality have been chosen (p).

iv. Project the bag-of-word features of labeled data (train/test) onto the reduced space, obtaining p new features.

Figure III.4 illustrates the steps above. As this feature space is obtained from the SMA matrix we will refer to the new features as SMA features.

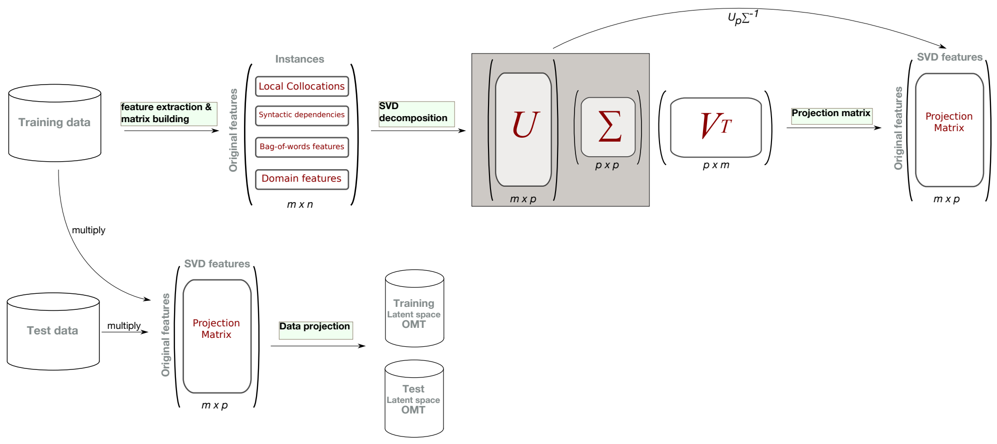

Figure III.5: Summary of the whole process of extracting the SVD features from OMT matrix: OMT features.

This technique is very similar to previous work on SVD (Gliozzo *et al.*, 2005). The dimensionality reduction is performed once, over the whole unlabeled corpus, and it is then applied to the labeled data of each word. The reduced space is constructed only with terms, which correspond to bag-of-words features, and thus discards the rest of the extracted features. Given that the WSD literature has shown that all features, including local and syntactic features, are necessary for optimal performance (Pradhan *et al.*, 2007), we explored other alternatives to construct the matrices, as explained in the following section.

The use of this type of features are reported in the following chapters: Chapter IV, Chapter V and Chapter VI.

III.3.3.2 One matrix per target word

The WSD literature has shown that complex and rich features are necessary for an optimal performance (Pradhan *et al.*, 2007). We propose an alternative method for applying SVD based on the idea of using all features (not only bag-of-words as in SMA) and treating each occurrence of the target word as documents. that is, instead of terms we use all extracted features, and instead of documents we use occurrences of the target word. We will call this feature-by-instance matrices **one matrix per target word** (OMT). Note that this variant performs one SVD process for each target word separately, hence its name. As in previous matrix the goal is twofold:

- Find correlations among features that show similar occurrence pattern in training set. The idea is to collapse in one dimension those features that manage similar information.
- Increase the overlap among the training and testing vector and, this way, deal with the data sparseness problem.

Contrary to SMA, we construct one matrix per target word based on the training set. So the method for each word is as follows:

i. Construct a corpus with occurrences of the target word. The corpus can include labeled training corpus, but given the fact that we do not use sense labels in the process, we can also add large amounts of unlabeled data (see Section III.3.3.3 for further details).

ii. Extract features from the context of each occurring instance.

iii. Build the feature-by-example matrix and, if desired, apply a weighting scheme.

iv. Decompose it with SVD, select the desired number of dimensions, and construct the projection matrix ($U_p\Sigma_p^{-1}$).

v. Finally, project all labeled training and test instances into the reduced space, thus obtaining p new features.

Figure III.5 illustrated all the steps explained above. We call the newly induced features OMT features.

The use of this type of features is reported in Chapter IV, Chapter V and Chapter VI.

III.3.3.3 SVD with unlabeled data

The knowledge acquisition bottleneck is a critical problem in WSD. If we were to apply OMT over the training instances, the resulting matrix would only contain tens of examples, and the correlations found in the data might not be reliable. Given the fact that the sense (label) of the instances is not used in the SVD process, we can use unlabeled data to have a larger matrix for each word, and hopefully obtain better correlations in the reduced space. Depending on the experiment and the domain we have used different unlabeled corpora to get large amounts of unlabeled instances, and thus augment the feature-by-instance matrix M into augmented M'. In our experiments we have tested different amounts of unlabeled data. We call this process **background learning**. A similar idea is used in (Zelikovitz and Hirsh, 2001) with Latent Semantic Indexing for Text Categorization. Once we have done the SVD decomposition of M' we obtain the new U' and Σ'_p, we project training and testing instances as in Equation (III.17) and can thus proceed to apply any ML method.

In the case of SMA matrices, the starting point is already untagged corpora, where we do not have any sense tagged. In this case, we also experimented with different sources and sizes of corpora.

Chapter IV will focus on the use of unlabeled data without any domain adaptation. Chapter V and Chapter VI will report the effect of domains and sources of the unlabeled data in semi-supervised domain adaptation and supervised domain adaptation, respectively.

III.3.4 Prior Work on SVD for WSD

Although SVD is a well-known technique and widely used in several disciplines out of the NLP area such as Psychology, Image Retrieval, Signal Processing, etc. It is has also been widely used in Information Retrieval, Text classification and Term Similarity, but it's application to WSD is more rare. Below, we will present some relevant literature that use SVD for WSD, but will also include work in other areas.

III.3.4.1 A kernel PCA method (Wu *et al.*, 2004)

Wu *et al.* (2004) presented a WSD system based on Kernel Principal Component Analysis (KPCA), which is a dimensionality reduction technique. They outperformed the best systems at the date in the Senseval-2 data (cf. Section II.4) obtaining %65.7 of accuracy. Their results were statistically significant at 0.10 level according to bootstrap resampling method.

They built a kernel which performed PCA over a nonlinear implicit function. PCA can be understood as the equivalent to apply SVD on a covariance matrix of data. They claimed that the nonlinear principal components analysis applied stronger generalization biases, taking combinations of predictive features into account. In (Su *et al.*, 2004), they used large amounts of unlabeled data in order to improve the results in Senseval-2.

III.3.4.2 Complex kernel combination (Gliozzo *et al.*, 2005)

This work combines various knowledge sources into a complex kernel combination. The kernels are used by a SVM algorithm. Among the kernels, Gliozzo *et al.* (2005) include the LSA kernel, which uses SVD to reduce the space of the term-to-document matrix. They computed the similarity between train and test instances using a mapping to the reduced space as an implicit function of the LSA kernel (similar to our SMA method in Section III.3.3.1).

They report state-of-the-art performance on a number of languages in the Senseval-3 lexical-sample datasets. An early version of the method was used in the Senseval-3 lexical-sample competition (Strapparava *et al.*, 2004) (cf. Section II.4.3). Our present work differs from theirs in that we propose an additional, more effective, method to use SVD (the OMT method), and that we focus on domain adaptation.

III.3.4.3 Alternating Structure Optimization (Ando, 2006)

Ando (2006) applied Alternative Structured Optimization (ASO) to WSD. ASO tries to take advantage of all labeled examples (irrespective of the target word) for learning, using labeled training examples for other words. For such purpose, she used a multi-task learning framework, creating m predictors (one for each target word) and finding the structure which represent the relations with respect to the other problems. This structure is found applying SVD to feature-by-predictor matrices, in contrast to our use of SVD over term-by-documents, or features-by-examples.

She first trained one linear predictor for each target word, and then performed SVD on 7 carefully selected submatrices of the feature-by-predictor matrix of weights. In order to make the selection, the author focuses on the feature type and the PoS of the predictors. The system attained small but consistent improvements (no significance data was given) on the Senseval-3 lexical sample datasets using SVD for multi-task learning and unlabeled data.

III.3.4.4 A multiclassifier in a reduced space (Zelaia *et al.*, 2008)

This work sets up several k-NN classifiers, and combine them using a Bayesian voting scheme. They used OMT feature space, as inspired by this dissertation. Once they reduced the space, they implement a bagging approach, and by random resampling they obtain a set of k-NN classifiers.

Their system was tested in SemEval-2007 lexical-sample data set and they obtain 85.65% of F-score. Among the number of parameters (number of classifiers, size of training set, number k nearest neighbor) they analyzed the effect of SVD dimensions. They concluded that the best dimension was set as half of the dimensions according to the number of training instances. Note that, in SemEval, all the words in training set not do share the same amount of examples, so that each word in the test has a different dimension reduction.

III.3.4.5 SVD feature selection for taxonomy learning (Francesca and Zanzotto, 2009)

Although this work does not focus on WSD, it is highly related since some kind of semantic disambiguation is needed in order to learn taxonomic relations. As a learning model they use probabilistic learning model introduced by Snow *et al.* (2006), and they use SVD as unsupervised feature selection method for their logistic regression classifier.

Their evaluation consists in determining how well their model can replicate an existing taxonomy (a portion of WordNet 1.5). The use of SVD is beneficial, obtaining up to 55% of accuracy.

III.3.4.6 Works from other areas

Zelikovitz and Hirsh (2001) introduce the notion of background learning for Text Classification (TC). They claim that LSI can suffer when there is little data to train. For example, many words occur only once in small datasets, and thus limit the power of LSI to create a proper model. The solution is to use large amounts of unlabeled data in order to get richer and more reliable patterns from text. The intuition is tested on four datasets for TC: Technical papers, Web page titles, WebKB and 20 Newsgroup. In all the dataset, except WebKB, LSI with background outperforms default version of LSI. In WebKB both approaches show similar performances. In addition, the use of background text show robustness, since performance on different amount of training data vary less than without unlabeled data. They conclude that the use of unlabeled data seems to be useful when there is lack of training data.

(Kim *et al.*, 2005) presents an extensive survey about dimensionality reduction in TC. They try several dimensionality reduction methods: SVD, Centroid-based algorithm for dimension reduction of clustered data, and Generalized-SVD. The last two methods try to maintain the clustering structure, but the latter maximizes the scatter between clusters while minimizes the scatter within clusters by defining two matrices and applying a dimensional reduction technique. The reduction techniques were applied over three ML algorithms: VSM, SVM and k-NN (cf. Section III.2.1). Surprisingly, the best improvement is obtained when k-NN when generalized-SVD is performed. Although SVM algorithm outperform the rest of the classifiers, the improvement is marginal when SVD is applied. Similarly, in (Zelaia *et al.*, 2005) k-NN obtains the best accuracy when dimension reduction is performed, and again SVM algorithm decrease its performance. According to this dissertation, clustering the training data would increase the bias of the model, and this could be harmful for the domain adaptation strategies.

Pereira and Gordon (2006) presents a novel classification method which combines dimensionality reduction with SVD and optimization of a single learning objective. They present an efficient algorithm which optimizes these two objective in a single optimization process. They recall that usually SVD computes the decomposition without reference to the label of training data. They introduce the decomposition procedure as a component inside the optimization problem, giving some weight regarding the loss function. The optimization is carred out by a sequential optimization method. The Support Vector Decomposition Machine (SVDM) is tested in fMRI analysis and the results show that this method outperforms the rest of the two phase approaches.

In our opinion, SVDM is not a useful strategy for domain adaptation, since SVDM pays attention on the classification problem, and herein will be biased to training distribution, which differs from the test data.

III.4 Ensembles of classifiers

The combination paradigm, known as ensembles of classifiers, is a very well-known approach in the ML community. It helps reduce variance and yields more robust classification results. The important point is that the errors produced by the combined classifier are not as correlated as those produced by the single classifiers. (Dietterich, 1997) provides interesting insights motivating classifier combination.

The most common combination schemes are based on weighted voting, where depending on the reliability of each classifier, the given vote is weighted. Nevertheless, this weighing scheme can show overfitting when the combined classifier is tested on domains different from the training domain. We propose a combination scheme that where the bias of the single classifiers is removed, but still keeping the vote weighted. This is possible due to the k-NN classifiers properties.

We also test kernel combinations. Kernels give the chance to combine different knowledge sources (different feature spaces) in very elegant and simple ways (Cristianini and Shawe-Taylor, 2000).

In this dissertation we deal with the sparseness and redundancy of the data by means of combining classifiers with different feature spaces. We have shown that the richer feature space, better are the results (Agirre and Martínez, 2004a). Large feature spaces tend to have highly redundant and heterogeneous features. Regarding heterogeneity, splitting the feature space might allow the learning algorithm to better capture the patterns in the data. Obtaining more coherent feature spaces we could in principle avoid the noise created by redundant information. We tested three possible improvements:

- Apply SVD to find correlations in the feature space.
- Use unlabeled data from different domain sources for background learning.
- Separate the feature space and train different voting classifiers. Note that in the latter voting scheme the SVD features are introduced.

The following sections will describe the two combination schemes proposed in this dissertation. Section III.4.1 describes k-NN combination, and Section III.4.2 shows our approach for kernel combination.

III.4.1 k-NN combination

Our k-NN combination method takes advantage of the properties of k-NN classifiers and exploit the fact that a classifier can be seen as k points (number of nearest neighbor) each casting one vote. This makes easy to combine several classifiers, one for each feature space. For instance, taking two k-NN classifiers of $k = 5$, C_1 and C_2, we can combine them into a single $k = 10$ classifier, where five votes come from C_1 and five from C_2. This allows to smoothly combine classifiers from different feature spaces.

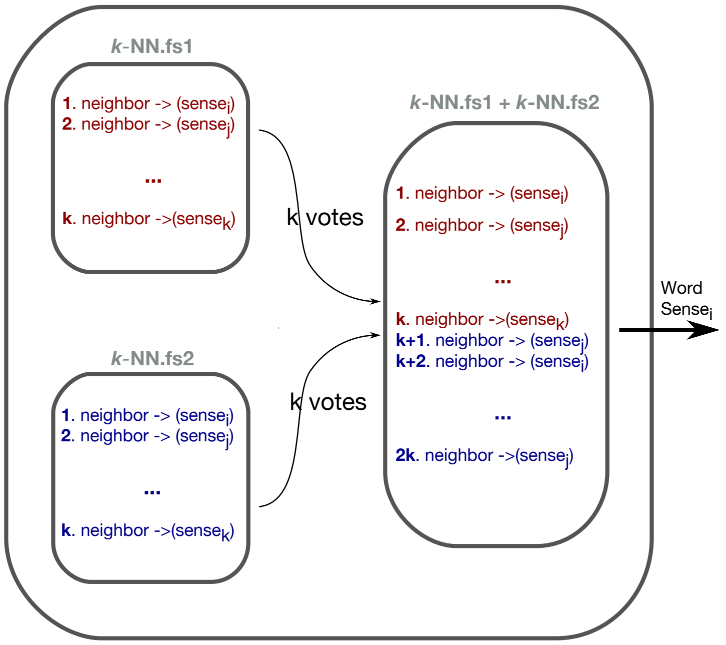

Figure III.6: How two k-nn systems trained on different feature space can be combined. Each system casts k votes and final decision is taken among $2k$ votes.

Figure III.6 shows how two k-NN systems, each trained in a different feature space, can be combined. Every vote (one of the k neighbor) is weighted depending on its classifier and space, and each classifier will cast k votes. The final decision is taken among all the casted neighbors. The Figure shows a combination of two systems, but equally can be generalized to n k-NN systems.

In order to combine them we firstly weighted each vote based on the cosine similarity of each selected neighbor. Then we noticed that each features space has its own similarity scale and it could introduced some kind of bias through certain classifier. For the rest of the experiments, in order to keep away from this situation, we decided to weight each vote by inverse ratio of its position in the rank of the single classifier, $(k - r_i + 1)/k$, where r_i is the rank. The rank weighting methods alleviate bias problem and maintains the importance of some neighbors regarding to others less similar.

In this work we built various single k-NN classifiers trained on a number of features spaces. The features spaces are based in three main featured presented up to this point: OMT, SMA and the original features. We will show through the dissertation that this way to combine feature space is an effective and robust one.

Combination of k-NN classifiers is a relatively unexplored area. Related to WSD task, for instance, the JHU-English system (Yarowsky *et al.*, 2001; Florian *et al.*, 2002), which used a voting scheme, obtained the best performance at English lexical sample task in Senseval-2. The main conclusions of their study was that the feature space had significantly greater impact than the algorithm choice, and that the different algorithms help to construct significantly a more robust WSD system.

Kohomban and Lee (2005) showed in a different WSD task that building separate k-NN classifiers from different subsets of features and combining them works better than constructing a single classifier with the entire feature set. Their combination was based on the weighted single voting, and they used held-out development data set for adjusting classifiers weights.

Another approach is presented 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, and simple collocations extracted from the context. A memory based learning algorithm is then applied to classify new examples.

An interesting method to create a multiclassiefier is proposed by Zelaia *et al.* (2008). Authors present a multiclassifier based on obtaining multiple training sets by ramdon subsampling and training k-nn classifier on each sampled training test. The combination is done by applying a Bayesian voting scheme, where the confidence of each single classifier is calculated in the training phase.

Not related to the WSD task, Bay (1999) describes MFS (Multiple Feature Subset), a combination of NN (nearest neighbor) algorithms that classifies using simple voting from NN classifiers, each having access only to a random subset of feature. As we did, the author built less accurate single classifiers (each classifier making independent errors) and joined them into a unique classifier. The author tried to build less correlated feature subsets in order to make independent errors. The experiments were done in several datasets from the UCI Repository. Related to this, Dietterich (1997) claims that splitting features only works when the feature space is highly redundant. We implemented the proposal from Bay (1999) in order to compare to ours.

III.4.2 Kernel combination

In Section III.2.1.5 we have introduced some aspects of kernels, describing their essential properties. This section will describe our proposal to combine the obtained feature spaces via kernels.

In previous Section III.3.3 we explained how to extract learning features by mapping with SVD matrices. In this section introduce how to use those mappings in a implicit and simple way by using kernels.

As explained previously, the basic idea is to find a suitable mapping function (ϕ) in order to get a better generalization. Instead of doing this mapping explicitly, kernels give the chance to do it inside the algorithm: $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$. This way, we can very easily define mappings representing different information sources and use this mappings in several machine learning algorithm. In our work we used SVM.

In order to exploit the properties of the different mapping functions defined in Section III.3, we defined three individual kernels. Each kernel corresponds to the previously explained feature spaces (OMT, SMA, original features). We have taken advantage of the SVD decomposition and its resulting mapping matrix and have defined them as an implicit mapping function. Afterward, we define a kernel combination schema in order to learn together the three feature spaces.

Firstly, we define the **original feature kernel** (k_{orig}). Our aim is to use the information from the original features. This way, the mapping function can be considered as identity function, $\phi : \mathcal{X} \rightarrow \mathcal{X}$, and the kernel may be defined as:

$$k_{orig}(\mathbf{x_i}, \mathbf{x_j}) = \frac{\langle \mathbf{x_i} \cdot \mathbf{x_j} \rangle}{\sqrt{\langle \mathbf{x_i} \cdot \mathbf{x_i} \rangle \langle \mathbf{x_j} \cdot \mathbf{x_j} \rangle}}$$

The denominator part is used to normalize and avoid any kind of bias in the combination.

Next we define the OMT **kernel** (k_{omt}) and SMA **kernel** (k_{sma}) by using OMT and SMA projection matrices, respectively (cf. Section III.3), in order to defined mapping functions. We define OMT mapping function as

$\phi_{omt} : \mathbb{R}^m \to \mathbb{R}^p$, where m is the number of the original features and p the new dimensionality into the projected space by OMT matrix. For the SMA mapping function (ϕ_{sma}) we define a similar function, but instead of using OMT, SMA matrices are used. So both kernels, k_{omt} and k_{Sma}, may be defined as follow[4]:

$$k_{svd}(\mathbf{x_i}, \mathbf{x_j}) = \frac{\langle \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j}) \rangle}{\sqrt{\langle \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_i}) \rangle \langle \phi(\mathbf{x_j}) \cdot \phi(\mathbf{x_j}) \rangle}}$$

Finally, we define the kernel combination schema capturing the properties of each mapping function and generalizing together as follows:

$$k_{comb}(\mathbf{x_i}, \mathbf{x_j}) = \sum_{l=1}^{n} \frac{k_l(\mathbf{x_i}, \mathbf{x_j})}{\sqrt{k_l(\mathbf{x_i}, \mathbf{x_i}) k_l(\mathbf{x_j}, \mathbf{x_j})}}$$

where n is the number of single kernels explained above, and l the index for kernel type. Note the combination of these kernels into a single one keeps the $\mathbf{K}$ kernel matrix (Gram matrix) as positive semi-definite.

[4]In order avoid repetition in formulas we define only once as k_{svd}. It is enough to substitute ϕ by ϕ_{omt} or ϕ_{sma} to obtain k_{omt} and k_{sma}, respectively.

CHAPTER IV

Combination of feature spaces and unlabeled data with SVD for WSD

In this chapter we explore three different improvements to state-of-the-art systems: 1) using Singular Value Decomposition in order to find correlations among features, trying to deal with sparsity, 2) using unlabeled data from a corpus related to the evaluation corpus, and 3) splitting the feature space into smaller, more coherent, sets. Each of the proposals improves the results, and properly combined are able to improve the state-of-the-art results for the different Senseval and SemEval dataset (lexical-sample and all-words). The analysis of the results provides further insights and possibilities for the future.

IV.1 Introduction

ML methods that rely on tagged text have to face data sparseness, specially in WSD, where only a small amount of tagged data is available (cf. Section I.4). In addition, supervised WSD systems degrade when the domain of the train and test texts differs (we shall deal with this in the following chapters). The impact of the above problems is exemplified by the frustrating handful of systems which are able to beat the simple MFS baseline in all-words tasks (Snyder and Palmer, 2004; Pradhan *et al.*, 2007).

As explained in Chapter III, we will tackle the sparsity problem using SVD, unlabeled data, and combinations of splits of the feature space. We will show that each of the modifications in the feature space improves the results, and properly combined they obtain state-of-the-art results. The analysis of the results will provide further insights and possibilities for the future.

Several experiments have been performed in a number of datasets. This chapter reports results on the lexical-sample and all-words tasks of Senseval-2, Senseval-3 and SemEval-2007 datasets. Some experiments cover different goals, and thus the order of the experiments, as reported, does not follow a chronological order. Figure IV.1 organizes all the experiments performed in this chapter. The figure details the goals for each dataset and which techniques have been used.

In the first set of experiments we will report the results obtained in Senseval-3 lexical-sample using SVD-OMT features, unlabeled data to augment the SVD matrices, and splits of feature sets. Similarly, in the second set of experiments we will focus on the features obtained from SMA matrix, where we used unlabeled data and a split of the feature set in smaller spaces. In this case, we set the parameters in Senseval-2 lexical-sample dataset, and test in Senseval-3 lexical-sample and all-words. Finally, in the third set of experiments we will report our participation in SemEval-2007 lexical-sample and all-words. In this dataset we tried to confirm our findings in the previous experiments.

The chapter is structured as follows. Section IV.2 will organize the experiments and remark the explicit goals, including the feature set and ML methods used, whereas Sections IV.3 to IV.5 show the results achieved in each experiment setting. Finally, Section IV.6 draws the conclusions and the future work.

IV.2 Experimental settings

The aim of this section is to introduce the experiments we performed in the current chapter and link each dataset with a set of experiments (see below). In order to facilitate the reading of the results and the conclusion we will remark the goals of each setting (Figure IV.1 summarizes the experiments, and their goals for this Chapter). The Section is organized as follow, first will describe the target datasets for our experiment. Next section will list the learning features, and the ML algorithms tested in this chapter, including the applied combinations, used the experiments.

First set of experiments

- **Goal**: to prove the usefulness SVD-OMT application (see Section III.3.3.2) , unlabeled data, and separation of feature spaces.
- **Dataset**: Senseval-3 lexical-sample task.

Second set of experiments

- **Goal**: Analysis on SVD-SMA application (see Section III.3.3.1), unlabeled data and separation of feature spaces.
- **Datasets**:
  - Senseval-2 lexical-sample: Optimization of parameters.
  - Senseval-3 lexical-sample: Test set.
  - Senseval-3 all-words: Test set.

Third set of experiment

- **Goal**: Confirms of findings of SVD-OMT, SVD-SMA and other feature space combination, and participate on the SemEval-2007 competition.
- **Dataset**:
  - Semeval 2007 lexical-sample.
  - Semeval 2007 all-words.

Figure IV.1: A brief summarization of the experimental settings: The goals and used components for such purpose.

IV.2.1 Datasets

Senseval-2 lexical-sample was used for parameter optimization. Specially, we selected the best SVD dimension and combinations setting. Regarding k-NN, previous experiments (Senseval-3 lexical-sample dataset) showed that 5 was a good choice, and therefore all the following experiments were carried with a k set to 5. VSM does not need any optimization, since it has not any parameter, and finally, we have used SVM optimized in a previous experiments (Senseval-3 lexical-sample). The results on this test set are described in Section IV.4.

Senseval-3 lexical-sample has been used in two scenarios. In the first our goal was to prove the usefulness of the SVD over one matrix per target word technique (OMT; see Section III.3.3.2). In the second experiment we wanted to confirm the results over Senseval-2 lexical sample.The results with OMT are shown in Section IV.3, and for SMA in Section IV.4.

Senseval-3 all-words has been used to confirm findings from Senseval-2 lexical-sample. The best SMA-SVD dimensions, the optimum amount of unlabeled data and best combination were we chosen according to the results in Senseval-2. The details on the test set are described in Chapter II, and we show the results in Section IV.4.

In Section IV.5 we will describe our participation in **SemEval 2007 lexical-sample**. As in the rest of the datasets, we focused on two SVD applications (OMT and SMA), the split of features set and k-NN combinations. In the same section, we describe our participation on the **all words task of SemEval 2007**.

The **British National Corpus** (BNC) was used to obtain the unlabeled examples to augment the OMT and SMA matrices in all experiments.

IV.2.2 Learning Features

The set of features is split in two. On the one hand we had original the features, those usual in WSD systems, which are directly extracted from the target corpus. They comprise **local collocations**, **syntactic dependencies**, **bag-of-words** and **domain features** (cf. Section III.1). Note that domain features only have been used the third set of experiments. On the other hand we have the SVD features, which are induced from the original features via dimensionality reduction. We used both OMT and SMA features (Section III.3.3). OMT were used on the first and third set of experiments, and SMA on the second and third sets.

IV.2.3 Learning methods

We used three well known classifiers in this chapter: Support Vector Machines (SVM) , k-Nearest Neighbors (k-NN), and the Vector Space Model (VSM). All these methods are more detailed explained in Section II.3.2. Regarding SVM we used linear kernels in SVM-Light (Joachims, 1999) implementation. We estimated the soft margin (C) for each word using a greedy process in a preliminary experiment on the source training data using cross-validation.

k-NN, a memory based learning method, the similarity among instances was measured by the cosine of their vectors. The test instances were labeled with the sense obtaining by the maximum the sum of the weighted vote of the k most similar contexts. k was set performing 3-fold cross-validation in the training data.

Finally, for the VSM each occurrence context is represented as a vector, and each sense in the training data is characterized by a centroid vector. The test vector is sense-assigned by the closest centroid. The similarity function is computed by the cosine formula.

Systems combination. We used combination of k-nn systems in order to combine different feature spaces. As explained in Section III.4.1, we built several single k-nn classifiers trained each in split spaces or in latent SVD spaces. In order to combine them each single k-NN classifier cast k neighbors, being each one vote. Depending on the experiment we weight each vote either by the cosine similarity computed in its space or by the inverse ratio of its position in the rank of the single classifier, $(\frac{k-r_i+1}{k})$, where r_i is the rank.

IV.3 1st experiments: SVD-OMT, unlabeled data and features split

In this set of experiments we wanted to test the usefulness of SVD for WSD task. Specially, we focused on OMT in order to extract the new learning features. In addition to that, we augmented the matrices with unlabeled data and split the features. Finally, we combined all the new features relying on different k-NN systems, each one trained on a different feature space.

The current section is organized as follow. First, we will present the results of the baseline methods. Next sections will describe the improved system based on the previously shown improvements (SVD, unlabeled data

Classifier	**Recall**
k-NN k=5	67.7
k-NN k=4	67.4
SVM	62.3
VSM	68.0

Table IV.1: Results for baseline classifiers in 3-fold cross-validation (Senseval-3 training set).

Classifier	**Recall**
k-NN k=5	70.5
SVM	71.2
VSM	71.5

Table IV.2: Results for baseline classifiers in the Senseval-3 lexical-sample test set.

and feature split), their optimization and results. The results of combination will be shown in Section IV.3.3, and finally we will discuss the achieved goals.

IV.3.1 Results of baseline methods

Initially we test the performance of k-NN, SVM and VSM trained on the original features and obtained the baseline results . VSM has no parameters, but k-NN needs to find an optimal k (number of neighbors) and SVM allows to optimize the "soft margin". We used 3-fold cross-validation on the Senseval-3 lexical-sample training set. For k-NN we only tried two values: $k = 5$ and $k = 4$. For SVM we used the "soft margin" value obtained by cross-validation on greedy process from other experiments.

Table IV.1 shows the results from cross-validation. We can see that the results of VSM and k-NN are very similar, with VSMoutperforming k-NNfor 0.3 points, and SVMperforming much lower[1]. For the rest of the experiments, we set $k = 5$ for all uses of k-NN. The results on the test set are shown in Table IV.2, with VSM increasing its advantage over k-NN and SVM in the middle of both.

[1]This could be because we used the wrong C estimation

IV.3.2 Improved systems

Now, we will show the results for the improved systems. We tested the previously shown baseline systems on the OMT features and on the split features spaces.

Splitting feature space

As seen in Section IV.2.2, WSD uses a high number of heterogeneous features. All the methods used in these experiments are based on geometrical properties of the feature space. If we split the problem into more coherent feature sets, the classification algorithms should find easier its way in such a simple space. We can thus build separate classifiers for each set of features, and hopefully obtain better results.

In order to test this hypothesis we split the features in two subsets:

- **Topical features**: Comprising the bag-of-word features.
- **Local features**: Comprising the local collocations and the syntactic dependencies.

Parameter setting for SVD

SVD needs to set several parameters which can affect the performance. In order to set those parameters we ran several preliminary experiments using SVD coupled with k-NN using 3-fold cross-validation as before. In the rest of the chapter, SVD was performed using the following parameters:

- Number of desired **dimensions**: We tried with 100, 200, 300, 500 and 1000 dimensions, and the best performance was obtained with 200 dimensions.
- **Weighting scheme** for the frequencies in the feature-by-instance matrix: We tried different classic schemes, including local weighting formulas such as term frequency (tf), *log* and *binary* ($min\{tf_{ij}, 1\}$), and global measures like *inverse document frequency* (*idf*) and *entropy*. For these set of experiments we used *log* and *entropy* weighting scheme, replacing the term frequency in the cells of the matrix, $t_{ij} \in M$, by $log(t_{ij}) \cdot entropy(i)$.

Classifier	**Recall**
k-NN k=5	67.7
SVM	62.3
VSM	68.0
k-NN-OMT k=5	69.8
SVM-OMT	61.2
VSM-OMT	63.9

Table IV.3: Results for *k*-NN and VSM with SVD in 3-fold cross-validation (Senseval-3 training set).

k-NN ($k = 5$)	**Recall**	**diff.**
plain	67.7	—
local+topical	69.4	+1.7
OMT	69.6	+1.9
OMT (25% BNC)	69.2	+1.5
OMT (50% BNC)	69.6	+1.9

Table IV.4: Improved *k*-NN classifier in 3-fold cross-validation (Senseval-3 training set). Plain stands for baseline *k*-NN.

- **Threshold** for global frequency (g): After building the matrix we can remove features that are very common (the less informative). We tried with different thresholds, and finally we chose to accept all features ($g = 0$).

Improved results

Table IV.3 presents the results of doing SVD, and then applying VSM, SVM and *k*-NN on the reduced space. We can observe that only *k*-NN improves performance, with VSM and SVM getting lower results. These and other prior experiments motivated us to only use *k*-NN on the improved systems.

Table IV.4 shows the results on the training set for the baseline *k*-NN systems, as well as all improvements explored. Plain stands for the baseline *k*-NN SYSTEM. The difference over the baseline system shows that all improvements were positive, raising from 1.5 to 1.9 the performance of the baseline. Still, there is no improvement observed when introducing unlabeled data into SVD (OMT + 25% of the BNC and OMT + 50% of the BNC)

Classifier	Recall	diff.
plain	70.5	—
local+topical	70.8	+0.3
OMT	70.7	+0.2
OMT (25% BNC)	70.8	+0.2
OMT (50% BNC)	71.2	+0.7
VSM	71.5	+1.0
SVM	71.2	+0.7

Table IV.5: Improved k-NN classifier in the Senseval-3 lexical-sample test set. Plain stands for baseline k-NN. VSM and SVM results are also provided for comparison.

compared to using labeled data only (OMT in Table IV.4).

Table IV.5 shows the same data for all baseline systems (including VSM and SVM) on the test set. The improvement here is lower but consistent with Table IV.4. The only difference is that using 25% or 50% of the BNC as unlabeled data for SVD is better than not using labeled data. Table IV.5 also presents the results of the other two baseline systems, showing that all k-NN systems are below VSM and SVM. This motivated us to try to combine the k-NN classifiers.

IV.3.3 Results of combination methods

The results from the previous section show that the improved systems (Section IV.3.2) are able to increase the results of k-NN, but are still below our SVM and VSM baseline systems. The key observation here is that under each of the improved classifiers there is a slightly different feature space. All of them provide improvements, and are therefore able to generalize interesting properties of the problem space. If we are able to combine them properly, we might be able to further improve the results.

Here we exploited the fact that a k-NN classifier can be seen as k points casting each one vote, making easy a combination of several k-NN classifiers. In order to carry through the properties of each feature space, we decided to weight each vote by the cosine similarity of that point instead of the rank. We need to note that this combination method was also used in the previous section to combine the local and topical classifiers.

Table IV.6 shows the results over the training set. The following rows show the improved systems from the previous Section. Then the results of combining the algorithms two by two are shown, where each of the improved systems has been combined with the baseline k-NN system. The results show that all combinations attain better results than any of their components. We can also see that, in this setting, using unlabeled data (plain+OMT with 50% of the BNC) improves slightly over not using it (plain+OMT). Finally, the full combination of all the systems provides the best results. Note that for the full combination, we applied OMT (with only labeled data, plus 25% and 50% of the BNC) also to the local and topical classifiers.

The results on the test set, Table IV.7, confirm the cross-validation results. Note that unlabeled data makes a more significant improvement over plain+OMT. Below the combined system, Table IV.7 also shows our baseline systems, as well as the best system in the Senseval 3 competition and the best reported result to date. The full combination of our k-NN systems attains the best results of them all.

k-NN($k = 5$)	**Recall**	**diff.**
plain	67.7	—
local+topical	69.4	+1.7
OMT	69.6	+1.9
OMT (25% BNC)	69.2	+1.5
OMT (50% BNC)	69.6	+1.9
plain + local+topical	69.9	+2.2
plain + OMT	70.7	+3.0
plain + OMT (25% BNC)	70.7	+3.0
plain + OMT (50% BNC)	70.8	+3.1
full combination	**71.9**	+4.2

Table IV.6: Results for different combinations of k-NN classifiers in 3-fold cross-validation (Senseval-3 training set)

IV.3.4 Discussion on the experiments

The results show that we have been able to better model the feature space. SVD helps to find correlations among the features, and thus alleviate the sparse data and redundancy problems. Including unlabeled data provides very narrow performance increases, but combined with the other classifiers

Classifier	**Recall**	**diff.**
plain	70.5	—
local+topical	70.8	+0.3
OMT	70.7	+0.2
OMT(25% BNC)	70.8	+0.2
OMT(50% BNC)	71.2	+0.7
plain + local+topical	71.5	+1.0
plain + OMT	71.2	+0.7
plain + OMT (25% BNC)	72.3	+1.8
plain + OMT (50% BNC)	72.7	+2.2
full combination	**73.4**	+2.9
SVM	71.2	—
VSM	71.5	—
Best S3	72.9	—
Gliozzo *et al.* (2005)	73.3	—

Table IV.7: Results for different combinations of k-NN classifiers in the Senseval-3 lexical-sample test set. Plain stands for baseline k-NN. VSM and SVM results are also provided, as well as the best Senseval-3 system and the best result published to date.

it makes a difference. Splitting the feature space in two and combining the two spaces also improves the results. These improvements in isolation are not very large. In fact, the resulting k-NN systems are below our SVM and SVM baseline systems for the original feature set. But when we combine the k-NN algorithms over each of the feature spaces, we attain the best results to date in the Senseval-3 dataset.

We think that the reason explaining the extraordinary performance of the combination is that each of the changes in the feature space helps finding regularities in the data that k-NN could not find before. When we combine each of the simpler k-NN systems, we are looking for the word sense that is closest to the target instance in as many of the changed feature spaces as possible. All in all, the best system is the combinations of many features spaces. First we split the spaces in three: the original space (the whole set of features), the topical features (also know as bag-of-words features), and the local features (the rest of the features). For each split space we obtained 4 systems: (i) a k-NN trained on the features themselves (without SVD), and three systems based on OMT features: (ii) only with tagged corpora, (iii) adding 25% of the occurrences in the BNC, and (iv) adding 50% of the occurrences in the BNC. In total 12 systems were combined.

In (Gliozzo *et al.*, 2005) instead of splitting the feature space and then combining the classifiers, they used specialized kernels to model the similarity for each kind of features. They also used SVD but only for bag-of-words features, while we apply SVD to all features. The good performance of coupling k-NN and SVD are well known in the ML literature, e.g. (Thomasian *et al.*, 2005; Kim *et al.*, 2005; Zelaia *et al.*, 2008).

Although as a main conclusion we think that the real improvement is coming from the combination of the different k-nn systems, each trained on a different feature space, we have to admit that OMT paradigm, where each word has its own feature-by-instance matrices, is an expensive approach.

IV.4 2nd experiments: SMA features set and combinations

In the previous set of experiments we showed that SVD on OMT matrices significantly improves the results in WSD. Although OMT is useful, we consider that it is costly to be apply in an all-words scenario. That is why we decided to perform experiments using more cheaper, but less accurate, matrix. In this section will focus on the use of SMA matrices. In addition, we performed more experiments through combination of finer splits of features. We explored feature set combination for different WSD systems.

Regarding types of combination, on the one hand we compared the simple voting and k-NN combination, and on the other hand we compared our manually split feature set to randomly created sets.

We have set the best combination options using Senseval-2 lexical-sample, and then evaluated on two datasets (Senseval-3 lexical-sample and all-words). For the all-words test set we took SemCor as the training corpus.

From the previous set of experiments we know that there are many ways to split the feature set. For these experiments we tried finer set experiments. Following our criterion we tried to group regarding the kind of the feature type, trying build a coherent set of features. In total, we tried with 6 feature spaces related to original features (cf. Section III.1). Initially, we have the set with all the features, being the richest and most heterogeneous feature space. Another way for distributing them is to separate bag-of-words feature and rest features (Local collocations, salient bag-of-bigrams, and syntactic dependencies). To finish with original features, we try with the local collocations, syntactic dependencies and the bag-of-bigrams obtained from the context as described in (Pedersen, 2001). Regarding SVD features we performed the experiments with SMA features. We can summarize the used features in the following manner:

- **bow**: bag-of-words (open-class lemmas).
- **local**: local collocations.
- **sx**: syntactic dependencies.
- **bob**: bag-of-bigrams.
- **notbow**: all features except **bow**.
- **ehu**: all features.

IV.4.1 Development: Senseval-2 lexical-sample

We use the Senseval-2 lexical-sample (Kilgarriff, 2001) data for optimization purposed. We focused on finding the best combination set. For k-NN methods we saw, in previous experiments, that best k was 5. All the experiments were carried with a fixed k in 5. We have done an exploration for the best combination. The VSM does not need any optimization, as it has not any parameter, and finally, we have used SVM in a default mode, without any parameter optimization.

First, we will show the results for single classifiers in the different feature-spaces, and finally, we will describe the results for best combination.

Senseval-2 lexical-sample single classifiers

Table IV.8 shows the results, recall and precision, for each method trained and tested in several feature spaces (the extension of the classifier denote the feature space). The results show that the more feature types one throws into the algorithm, the better are the results as shown in (Agirre and Martínez, 2004a).

We can see richest feature space, *ehu*, is the best single classifier in every method, except for SVM which attains the same results for *ehu* and *notbow* feature distributions.

The VSM from the *ehu* feature space is the best method. It obtains 63.3 of recall in the Senseval-2 Lexical Sample.

Taking into account each of the feature sets, we note that generally the local collocations features (*local*) discriminate better than bag-of-words features (*bow*). Only in the case of VSM the *bow* features work better than *local* ones. The reason that syntactic dependencies (*sx*) and bag-of-bigrams (*bob*) do not work as well as local collocations and bag-of-words features might be the insufficient amount of feature instances that occur in the given context. But we see that they help when we throw into a more complex feature space (*notbow*).

Taking into account that SMA features are bag-of-words features projected onto a latent spaces, the table shows that SMA discriminates better than *bow* features. The association of higher order correlation works well for this dataset.

Senseval-2 lexical-sample combination optimization

In this section we report the results for the combination of the single systems in the previous section. As our focus is on the use of feature spaces, we tried all exhaustive combinations of feature sets, but always using a single learning method.

Table IV.9 shows the best combinations per learning method. For *k*-NN, *ehu+sma+notbow* yields the best results, for the *k*-NN combination. The best *k*-NN combination improve significantly over the single classifier (65.1 *vs* 62.4) and outperform the best system in competition (Yarowsky *et al.*, 2001).The combination which yields best results for SVM was *ehu+bow+notbow+bob*, and *ehu+bow+notbow+bob* was the best in the case of VSM. *k*-NN is the only one improving over the single methods, and attains the best results overall.

Table IV.9 also shows that the finer grained feature sets are not the best, and that using all features helps obtain better results.

Classifier	Coverage	Recall
knn.ehu	100	62.4
knn.notbow	100	60.4
knn.local	100	58.5
knn.sma	100	54.6
knn.bow	100	52.4
knn.sx	100	49.1
knn.bob	39.0	23.8 (60.8)
svm.ehu	100	61.0
svm.notbow	100	61.0
svm.local	100	60.9
svm.sx	100	56.1
svm.bow	100	55.8
svm.bob	100	51.9
vsm.ehu	100	63.3
vsm.notbow	100	57.2
vsm.bow	100	55.2
vsm.local	100	51.6
vsm.sx	100	41.6
vsm.bob	39.0	22.1 (56.6)

Table IV.8: Results for single classifiers in the Senseval-2 lexical-sample. Parenthesis for precision.

Classifier	Recall
knn.ehu+sma+notbow (knn-comb)	**65.1** ↑
vsm.ehu+bow+notbow+bob	63.0 ↓
knn.ehu+bow+notbow (single-vot)	62.2 ↓
svm.ehu+bow+notbow	61.0 =

Table IV.9: Results for best classifier combinations in the Senseval-2 lexical-sample. ↑ means *ehu* single system is worse, = means *ehu* system is equal, and ↓ is ehu system better. Coverage is 100% for all.

IV.4.2 Senseval-3 lexical-sample: Single and combined classifiers

Table IV.10 shows the results of the single and combined classifiers. As expected from the results in the previous section, the combinations for VSM and

SVM do not improve over the single results, but do improve over the k-NN system. The best results are for k-NN with the combined k-NN following closely.

Compared to more complex system described in Section IV.3 we are still 1.1 points below. Nevertheless, the obtained results were promising and encouraged us to keep on exploring the use of SMA features and feature sets combinations.

Classifier	**Recall**
knn.sma	63.9
vsm.ehu	**71.5**
knn.ehu	70.4
svm.ehu	69.2
knn.ehu+sma+notbow (knn-comb)	**72.3**
vsm.ehu+bow+notbow+bob	70.9
knn.ehu+bow+notbow (single-vot)	70.6
svm.ehu+bow+notbow	68.9
Best SL-3	72.9
OMT combination (Section IV.3.3)	**73.4**

Table IV.10: Results for best single and combined classifier in the Senseval-3 lexical-sample.Coverage is 100% for all.

IV.4.3 Senseval 3 all-words task

We use SemCor (Miller *et al.*, 1993) corpus for training, hand-tagged in WordNet 1.6. We have use the mapping from WordNet 1.6 to 1.7.1 (Daude *et al.*, 2000), because the Senseval-3 all-words task was hand-tagged using WordNet 1.7.1. In the case where target word has less than 10 instances in SemCor we have applied the most frequent sense.

We prepared a clean test set to make our systems results comparable with the official results from Senseval-3 all-words task. Taking into account that systems from Senseval-3 did not know the part-of-speech (PoS) of the target word, we have removed test instances where the PoS was wrongly assigned by the two best systems in the competition. We have also removed multiwords. After cleaning the test set comprises 1.819 instances.

As in the previous section, we used the parameter setting from Senseval-2 lexical-sample (cf. IV.4.1).

Classifier	Recall
knn.ehu+bow+notbow (knn-comb)	**68.5**
knn.ehu+sma+notbow (knn-comb)	**68.4**
svm.ehu	67.9
knn.ehu	67.8
knn.ehu+bow+notbow (single-vot)	67.8
GAMBL (best S3AW)	67.8
vsm.ehu	65.5

Table IV.11: Results for different systems in the Senseval-3 all-words task. Coverage is 100% for all.

Table IV.11 shows that the k-NN combination outperforms all the other systems, including the best system in Senseval-3 all-words task (GAMBL). We performed more combination (apart from the best in previous data sets) in order to confirm whether *ehu+sma+not-bow* was the best combination in the current dataset. Surprisingly, contrary to our expectation, *ehu+bow+notbow* combination obtained the best results, though the difference was marginal.

IV.4.4 Discussion on the experiments

We have shown that a simple method like a k-NN classifier can work as good as more complex, and a priori more powerful methods. Splitting the feature-space and then combining them into a single classifier obtains the best results up to date in the Senseval-3 all-words task.

The good results are due to the potential for k-NN classifiers to be combined. Rather than using "the one classifier one vote" paradigm, each classifier suggest the k closest instances (and their word senses) from their feature space and after that, the merged classifier sums them to decide the word sense.

We think that the reason of the good performance of the combination is that each of the changes in the feature space helps finding regularities in the data, which single k-NN could not find. When we combine each of the simpler k-NN systems, we are looking for the word sense that is closer to the target instance in different feature spaces. In other words, we are discriminating word senses from different point of view.

In order to compare our method of splitting the feature space with the proposal from (Bay, 1999) (cf. Section IV.4), we performed some additional experiments on the Senseval-3 lexical-sample dataset. Bay (1999) created random subsets of features and combined less accurate 1-NN classifiers to improve the classifier. Table IV.12 shows the results of applying a random split over original features in three feature sets, both using single voting over 1NN and our method for 5NN. We also tried using 1NN instead of 5NN, but the results are lower than combined system (last row).

Classifier	**Recall**
Random, 3 splits 1NN	65.7
Random, 3 splits 5NN	70.0
knn.ehu+bow+notbow 1NN	67.1
knn.ehu+bow+notbow 5NN	71.4

Table IV.12: Results for additional experiments on Senseval-3 lexical-sample.

IV.5 3^{rd} experiments: Semeval-2007

This last set of experiments describes our participation on lexical-sample and all-words WSD subtask of SemEval 2007 task 17. We applied a combination of different k-NN classifiers. As in the previous experiments each classifier manages different information sources, and also making the combination a powerful solution. Our main goal for this set of experiments have been to apply previous findings (first and second experiments) in a new datasets: We combined OMT, SMA and split original features on a number of k-NN.

It is important to note that the dataset for lexical-sample was annotated with the OntoNotes sense. Due to this, the comparison among the previous dataset becomes harder. Regarding all-words task occurrences were annotated with WordNet senses (version 2.1)

These experiments are organized as follows. Before we show the results, next section will describe performed feature splits. In Section IV.5.2 we will draw the results on the training data, for both lexical-sample ant all-words tasks. Finally, we will show the figures over the test data.

IV.5.1 Features splits for k-NN combinations

Our previous experience has shown that splitting the problem up into more coherent spaces, training different classifiers in each feature space, and then combining them into a single classifier is a good way to improve the results. Depending on the feature type (original features or features extracted from SVD projection) we split different sets of feature spaces. In total we tried 10 features spaces.

Following we will describe the features split based on **original features** and how we denoted them:

- **ehu**: Extracted all original features, including the domain features.
- **ehu-notdom**: All original features except domain features. This set of features have been used in the previous sections. In order to assess the contribution of the domain features we removed these type of features from the whole set.
- **local**: Those features comprising the local collocations and the syntactic dependencies. In other words: All the extracted original features except domain and bag-of-words features. This set have been previously shown.
- **topical**: The set of bag-of-words and domain features.This set have been previously shown.
- **bow**: Bag-of-word features.
- **dom**: Domain features.

For the SVD **features** we project onto reduced space several set of original features. Following we show the applied kinds of projection :

- OMT[**ehu**]: OMT matrix applied to the whole set of the original features.
- OMT[**local**]: OMT matrix with the **local** features.
- OMT[**topic**]: OMT matrix consisting of the **topical** features.
- SMA: Features obtained from the projection of **bow** features with the SMA matrix.

The votes in the combination process were weighted by the inverse ratio of its position in the rank $(k - r_i + 1)/k$, where r_i is the rank.

IV.5.2 Experiments on training data

This section will show the results obtained in Semeval-2007 on the training data, where we optimize some aspects of the WSD system. Due to dataset for lexical-sample was different, different sense-inventory was used in the tagging process, we decided to explore some of the parameter: The number of neighbors (k), the SVD dimension (d) and the optimal combination. First of all, results of the best single classifiers will be given, next we will show the improvement achieved when combination of several classifier are combined. We will describe a simple methodology to find an optimum ensemble of classifiers.

Regarding all-words, we used SemCor as a training data set and we based on the previous experiments (Section IV.4) to set the values of the parameters. The optimization focused on finding the best combination of feature spaces. Results of the single classifiers will be shown in order to test whether combination were useful or not.

Optimization for the lexical-sample task

For the lexical-sample task we only use the training data provided. We tuned the classifiers using 3 fold cross-validation on the SemEval lexical-sample training data. We tried to optimize several parameters: number of neighbors, SVD dimensions and best combination of the single k-NN systems. We set k as one of $1, 3, 5$ and 7, and the SVD dimension (d) as one of $50, 100, 200$ and 300. We also fixed the best combination. This is the optimization procedure we followed:

1. For each single classifier and feature set (see section IV.5.1), check each parameter combination.
2. Fix the parameters for each single classifier. In our case, $k = 5$ and $k = 7$ had similar results, so we postponed the decision. $d = 200$ was the best dimension for all classifiers, except OMT[topic] which was $d = 50$.
3. For the best parameter settings ($k = 5; k = 7$ and $d = 200$; $d = 50$ when OMT[topic]) make *a priori* meaningful combinations (due to CPU requirements, not all combination were feasible).
4. Choose the x best combination overall, and optimize word by word among these combination. We set $x = 8$ for this work, k was fixed in 5, and $d = 200$ (except with OMT[topic] which was $d = 50$).

Feature type	**Feature space**	**F-score**
Original features	ehu: $k = 5$	87.2
	ehu-notdom: $k = 5$	87.2
	local: $k = 7$	86.4
	topical: $k = 7$	83.0
SVD features	OMT: $d = 200$; $k = 7$	87.8
	OMT: $d = 200$; $k = 5$; 25% BNC	**88.1**
	OMT: $d = 200$; $k = 5$; 50% BNC	88.0
	SMA: $d = 200$; $k = 7$	82.4

Table IV.13: Result for the best k-NN classifiers for each single feature spaces. The experiments were performed using 3 fold cross-validation on SemEval lexical-sample.

The results obtained by single classifiers are shown in Table IV.13. The table is organized as follows. The first two columns denote the feature type and features space used by the k-NN classifiers, the third column shows the results in terms of F-score (note that our systems always answer, and thus the precision and recall are equal). We have omitted some results and only show the best system for each feature space. Focusing on the original features, the figures confirm that the more rich and complex is the feature space the better performance has the classifier. Contrary to our expectation domain features do not help improving the results, not at least in this data set.

The table shows that SVD on OMT improve considerably the results over the original space. We are able to improve 0.9 point in the best case. We can concluded that although unlabeled data help, only improved slightly. With respect to SMA, surprisingly obtained lowest results among all the classifiers.

Rank	**Combination**	**F-score**
1	ehu + topic + local+ OMT[ehu] + OMT[topic] + OMT[local]	88.8
2	ehu + ehu-dom + topic + local + SMA + OMT[ehu] + OMT[topic] + OMT[local]	88.7
3	ehu + topic+local + SMA + OMT[ehu] + OMT[topic] + OMT[local]	88.5
4	ehu-notdom+topic+local + SMA + OMT[ehu] + OMT[topic] + OMT[local]	88.5
5	ehu + ehu-notdom + topic + local	88.4
6	ehu-notdom + local + SMA	88.3
7	ehu + ehu-notdom + local + SMA	88.2
8	ehu + topic + local	88.1
	word-by-word optimization	**89.5**

Table IV.14: Result for the best k-NN combinations in 3 fold cross-validation SemEval lexical-sample.

Regarding combination, Table IV.14 shows the best eight results for 3 fold cross-validation in SemEval lexical-sample training corpus. The results show that on average the more different spaces you combine the better are the results, although there are some combinations that do not work well. The best combination (without taking to account word-by-word optimization) was the one which combines all the features, the topical and local features in both space, the original and the latent spaces. After the optimized each single word the performance increases 0.7 percentage points over the best combination, and comparing to the OMT best single classifier the results improve significantly in 1.4 points (2.3 point over k-nn trained on the original set features).

Regarding SMA, the results reveal that it is not as powerful as the OMT and the best combination does not contain this space. Nevertheless, the contribution on ensemble of classifier is positive: Second ranked system combine SMA features. Thus, we think that, due to its feasibility in all-words task, could be very useful for all-words WSD system.

Optimization for the all-words task

To train the classifiers for the all-words task we just used SemCor. In Section IV.4, we already tested our approach on the Senseval-3 all-words task. The best performance for the Senseval-3 all-words task was obtained with $k = 5$ and $d = 200$, but we decided to perform further experiments to search for the best combination. We tested the performance of the combination of single k-NN training on SemCor and testing both on the Senseval-3 all-words data (cf. Table IV.16) and on the training data from SemEval-2007 lexical-sample (cf. Table IV.17).

First of all, we will show the results of the single classifiers trained on SemCor and tested on SemEval-2007 lexical-sample. The results on Senseval-3 all-words have been shown in the previous set of experiments (cf. Section IV.4 and Table IV.11). Table IV.15 shows the results for each feature space. Again, the richest feature space (ehu) attains the best results, over 2.5 points on topical and local features. Concerning SVD, SMA features outperform significantly over the best system in original space. This confirms our hypothesis that SVD helps alleviating the data sparseness: Many words occur less than 10 times in SemCor.

Relating to the combinations, Table IV.16 and Table IV.17 show the some of the results obtained on Seneval-3 all-words and SemEval-2007 lexical-

Feature type	Feature space	F-score
Original features	ehu: $k = 5$	63.5
	topical: $k = 5$	60.8
	local: $k = 5$	60.6
SVD features	SMA: $d = 200$; $k = 5$	**66.8**

Table IV.15: Results for single k-NN classifier on SemEval-2007 lexical-sample, using Semcor as training corpus.

sample, respectively, taking them as test sets. The two tables are organized as the previous ones, where columns express the combined features and the attained score, respectively.

Regarding the results, note that tables IV.16[2] and IV.17 show contradictory results. While the same or similar combinations were performed in both data sets, they gave different responds in terms of F-score. On Senseval-3 all-words the best results was obtained by the system with all the feature splits (ehu + topical + local + SMA). Contrary, on Semeval-2007 lexical-sample the same set of feature splits was not able to improve over ehu + SMA neither SMA as single classifier (the best in this dataset).

Given that in SemEval-2007 lexical-sample the senses are more coarse grained, and Senseval-3 all-words should be more similar to test set on SemEval-2007 all-words, we decided to take the best combination on Senseval-3 all-words for the final submission. Therefore, the elected combination consisted of a set of all features (ehu), a set of topical features, a set of local features, and SMA features.

Combination	F-score
ehu + topical + local + SMA	68.9
ehu + local + notbow	68.5
ehu + local + SMA	67.9

Table IV.16: Results for the best k-NN combinations in Senseval-3 all-words, using SemCor as training corpus.

[2]notbow feature split have been was used for experimets described in Section IV.4.1

Combination	F-score
ehu + SMA	66.6
ehu +topical + local + SMA	66.4
ehu + local + SMA	66.1

Table IV.17: Results for the best k-NN combinations in training part of SemEval lexical-sample, using Semcor as training corpus.

IV.5.3 Official results

Table IV.18 shows the performance obtained by our system and the winning systems in the SemEval lexical-sample and all-words evaluation. On the lexical-sample evaluation our system is 2.6 lower than the cross-validation evaluation. This can be a sign of a slight overfitting on the training data due to the exhaustive searching of the best combination. All in all we ranked second over 13 systems.

Our all-words system did not perform so well. Our system is around 4.7 points below the winning system, ranking 5th from a total of 14, and 3 points above the baseline given by the organizers. This is a disappointing result when compared to our previous section on Senseval-3 all-words where we were able to beat the best official results. Note that the test set was rather small, with 465 occurrences only, which might indicate that the performance differences are not statistically significant. We plan to further investigate the reasons for our results.

Further details have been given in Section II.4.4, where we analized the SemEval-2007 edition.

Task	Method	Rank	F-score
LS	Best	1	88.7
LS	k-NN combination	2	86.9
LS	Baseline	-	78.0
AW	Best	1	59.1
AW	k-NN combination	5	54.4
AW	Baseline	-	51.4

Table IV.18: Official results for SemEval-2007 task 17 lexical sample and all-words subtasks.

IV.6 Conclusions

First set of experiments – OMT*, unlabeled data and feature split.* In this work we have explored feature modeling, trying to tackle sparse data, redundancy and heterogeneity in the feature set. We have proposed and evaluated three improvements: 1) using SVD in order to find correlations among features and deal with sparsity and redundancy, 2) using unlabeled data from a corpus related to the evaluation corpus in order to provide background knowledge, and 3) splitting the feature space into smaller, more coherent, sets. Each of the proposals improves the results for a k-NN classifier, and properly combined they provide we obtained one of the best results for the Senseval-3 lexical-sample dataset.

In the discussion we have argued that these improvements help to model better the feature space, which, coupled with a ML algorithm well suited for combination such as k-NN, explains the good results. This opens new feature modeling possibilities. In particular we think that using kernels to better model similarity for certain features might provide better results, as later shown in Chapter VI. On the other hand we have shown that unlabeled data helps, and later in the dissertation we explore which is the situation when the training and test data come from distinct corpora or domains.

Second set of experiments – SMA *features set and combinations.* This set of experiments explored the split of feature sets in order to obtain better WSD systems through combinations of classifiers learned over each of the split feature sets. Our results show that k-NN is able to profit from the combination of split features (contrary to VSM and SVM), and that simple voting is not enough for that. Instead we propose combining all k-NN subsystems where each of the k neighbors casts one vote. The reader will has noticed that we did not performs all the experiments with SVM and VSM algorithms. Due to our previous experience in the first set of experiments we decided not to perform any experiment on SMA features. Nevertheless, in the following chapters we will show some results for SVM on SMA.

The experiments explore different feature spaces by splitting a rich set of features into a smaller and less accurate, but more coherent, sets. The comparison with random splits suggest that a manual split based on the nature of the features is more productive than random splits.

We showed that SMA features behaved well through different datasets.

The experiments demonstrated that SMA and the combination with the original features finds different patterns in text, improving consistently the system performances.

We have performed a thorough evaluation on two datasets (Senseval-3 lexical-sample and all-words), having set the best combination options in a development dataset (Senseval-2 lexical-sample). The results for the all-words task were the best published at that date.

Third set of experiments – *Participation in Semeval-2007.* In these experiments we confirmed our findings in the previous experiments. First of all, we built a robust system based on a number of k-NN classifiers , each trained on a different feature spaces. The classifier was tested in lexical-sample and all-words tasks. Regarding lexical-sample, we saw that OMT features outperform the rest of the features, and that SMA features are useful in combinations. In all-words, we only experimented with SMA, obtaining good results.

Overall conclusions. In this Section we have explored several feature spaces, reporting results in different experiments for features induced from the SVD decomposition, with and without unlabeled data. We also explored different ways to combine these feature spaces.

We applied SVD to two different kinds of matrices. Our experiments show that the OMT technique to apply SVD compares favorably to SMA, which has been previously used in (Gliozzo *et al.*, 2005). Although constructing one matrix per target word (OMT) yields the best results, it is a relatively expensive process, so we did not apply it to the all-words setting. In the all-words experiment we only tested the single matrix for all (SMA) approach, with good results. The results show that our combined k-NN systems are state-of-the-art, specially in lexical sample settings, and that the induced features provide significant improvements. In the future, it would be interesting to apply OMT to all-words settings.

CHAPTER V

Semi-supervised domain adaptation

*In this chapter we will explore robustness and domain adaptation issues for Word Sense Disambiguation (*WSD*) using Singular Value Decomposition (*SVD*) and unlabeled data. We will focus on the semi-supervised domain adaptation scenario, where we train on the source corpus and test on the target corpus, and try to improve results using unlabeled data. Our method yields up to 16.3% error reduction compared to state-of-the-art systems, being the first to report successful semi-supervised domain adaptation. Surprisingly the improvement comes from the use of unlabeled data from the source corpus, and not from the target corpora, meaning that we get robustness rather than domain adaptation. In addition, we will study the behavior of our system on the target domain.*

V.1 Introduction

In the previous chapter we have studied the contribution of SVD. We also have showed that the use of unlabeled data might alleviate the data sparsity problem. And finally, we have showed the combination of various sets of feature space can be beneficial to improve the performance of the WSD systems. As a result, we have been able to outperform the state-of-the-art through several datasets. Now we will focus on domain shift problem: How we can use the previous contributions on domain adaption scenario.

In many NLP tasks we find that a large collection of manually-annotated text is used to train and test supervised machine learning models. While these models have been shown to perform very well when tested on the text collection related to the training data (what we call the **source** domain), the performance drops considerably when testing on text from other domains (called **target** domains).

As remarked in Section II.5, in order to build models that perform well in new (target) domains we usually find two settings (Daumé III, 2007): In the **semi-supervised** setting the goal is to improve the system trained on the source domain using unlabeled data from the target domain, and the baseline is that of the system trained on the source domain. In the **supervised setting**, training data from both source and target domains are used, and the baseline is provided by the system trained on the target domain. The semi-supervised setting is the most attractive, as it would save developers the need to hand-annotate target corpora every time a new domain is to be processed.

The main goal in this chapter is to use unlabeled data in order to get better domain-adaptation results for WSD in the **semi-supervised setting**. SVD has been shown to find correlations between terms which are helpful to overcome the scarcity of training data in WSD ((Gliozzo *et al.*, 2005), and showed in previous chapter). This chapter explores how this ability of SVD can be applied to the domain-adaptation of WSD systems, and we show that SVD and unlabeled data improve the results of two state-of-the-art WSD systems (k-NN and SVM). For the sake of this chapter we call this set of experiments the **semi-supervised domain adaptation scenario**.

In addition, we also perform some related experiments on just the target domain. We use unlabeled data in order to improve the results of a system trained and tested in the target domain. These results are complementary to the domain adaptation experiments, and also provide an upperbound for semi-supervised domain adaptation. We call these experiments the **target domain scenario**. This last scenario is closely related to the experiments performed in Chapter IV, since the data from training and test sets are coming from the same source. Note that both scenarios are semi-supervised, in that our focus is on the use of unlabeled data in addition to the available labeled data. Figure V.1 summarize all the experiments reported in this chapter.

The experiments were performed on a publicly available corpus which

Semi-supervised domain adaptation scenario: BNC $\rightarrow \mathcal{X}$

- **Goals**: Adapt a general purpose supervised WSD system from source (BNC) to target (SPORTS, FINANCE) domain.
- Explore the effects of distinct unlabeled data.
- **Unlabeled data**. Source domain: BNC; Target domain: SPORTS, FINANCE.
- **Features spaces**: SMA, OMT, and original features.

Target domain scenario: $\mathcal{X} \rightarrow \mathcal{X}$

- **Goals**: Train and test on the target domain (SPORTS, FINANCE).
- The upperbound for semi-supervised domain adaptation.
- **Unlabeled data**. Source domain: BNC; Target domain: SPORTS, FINANCE.
- **Features spaces**: SMA, OMT, and original features.

Figure V.1: The scheme of experiments in this chapter. Each scenario has defined its own goals and purposes.

was designed to study the effect of domain in WSD (Koeling *et al.*, 2005). It comprises 41 nouns closely related to the SPORTS and FINANCE domains with 300 examples for each. The 300 examples were drawn from the British National Corpus (Leech, 1992) (BNC), the SPORTS section of the Reuters corpus (Rose *et al.*, 2002), and the FINANCE section of Reuters in equal number (cf. Section II.1.2.2).

The chapter is structured as follows. Section V.2 is devoted to the experimental settings: learning methods and learning features will be reviewed. The experimental results are presented in Section V.3, for the semi-supervised domain adaptation scenario, and Section V.4, for the target scenario. Section V.5 presents the discussion and Section V.6 the conclusions and the future work.

V.2 Experimental settings

V.2.1 Learning methods

We used Support Vector Machines (SVM) and k-Nearest Neighbors (k-NN) (cf. Section III.2.1). Regarding SVM we used linear kernels implemented in SVM-Light (Joachims, 1999). We estimated the soft margin (C) for each feature space and each word using a greedy process in a preliminary experiment on the source training data using cross-validation. The same C value was used in the rest of the settings.

With k-NN the k nearest neighbors were calculated by using the cosine similarity, and we set k in 5 based on previous results (Chapter IV). When combining, the vote was weighted depending on its (neighbor) position in the ordered rank, with the closest being first.

V.2.2 Learning features

We relied on the learning features used in previous chapter (detailed description is given in Section III.1). The **original features** can be summaries as follows: **Local collocations**, **Syntactic dependencies** and **Bag-of-words features**.

As the same manner as in previous chapter we extracted the SVD **features** (cf. Section III.3) . We used both OMTand SMA features. We base on BNC, the FINANCE part of Reuters, and the SPORTS part in order to build matrix and extract features, depending on the domain we want to perform the adaptation.

Building Matrices

Due to high possibilities to combine the unlabeled, labeled data and build SMA and OMT matrices, we tried these possibilities: 1) use the train corpus alone; 2) add a corpus from the source domain (general domain in this case) to the train; and 3) add a domain-specific corpus from the same domain as the target corpus. These are the matrices which have been applied in the experiments:

- TRAIN: The matrix comprises features from labeled train examples alone. This matrix can only be used to obtain OMT features.

- TRAIN ∪ BNC: In addition to TRAIN, the matrix also includes unlabeled examples from the source corpus (BNC). Both OMT and SMA features can be obtained, since unlabeled data is used.

- TRAIN ∪ {SPORTS, FINANCE}: Like the previous, but using unlabeled examples from one of the target corpora (FINANCE or SPORTS) instead. Both OMT and SMA feature can be obtained.

Note that SMA features are always obtained from the unlabeled data (we do not use labeled data), but for simplicity, we report it in the same category as OMT features – when we use unlabeled data to extract them. For example, when we say that TRAIN ∪ BNC matrix is for SMA, it means that do not use the examples from training (only for projecting).

Based on previous work described in Chapter IV, we used 50% of the respective unlabeled corpora for OMT features, and the whole corpora for SMA. In Section V.3.2 we will report some experiments controlling the size of unlabeled corpora.

Dimensionality selection

An important parameter when applying SVD is the number of dimensions in the reduced space (p). We tried two different values for p (25 and 200) in the BNC domain. Once we set the best dimension for each feature space and ML algorithm, we keep the values for the sake of the experiments.

Table V.2 shows the effect of the dimensionality. We performed 3-fold cross-validation in order to select the best p dimension for each feature space and ML approach. The columns denote the classifier and the feature space used: the MFS column corresponds to the most frequent sense in the BNC dataset. k-NN-ORIG (SVM-ORIG) corresponds to performing k-NN (SVM) on the original feature space, k-NN-OMT (SVM-OMT) corresponds to k-NN (SVM) on the reduced dimensions of the OMT strategy, and k-NN-SMA (SVM-SMA) corresponds to k-NN (SVM) on the reduced dimensions of the SMA strategy (cf. Section III.3). The rows correspond to the matrix used for SVD (cf. Section V.2.2). Each OMT and SMA column has another sub-columns, which correspond to the dimension reductions (25 and 200 dimension). Note that some of the cells have no result, because that combination is not applicable, e.g. using the TRAIN ∪ BNC in the original space.

From the figures in Table V.2 we can conclude that when only labeled data is used (the figures on TRAIN row), 200 dimensions performs better for the

OMT in general terms. In the case of the k-NN-OMT both dimensions perform equal, but based on the results of previous chapter and the performance on SVM we consider 25 as best option for testing in the SPORTS and FINANCE.

When unlabeled data is used (TRAIN ∪ BNC row) for OMT 25 performs better in all the cases, besides, the difference are statistically significant. For k-NN-SMA the performance is very similar and there is no significant difference. Again, based on previous experiments (Chapter IV), we chose 200 dimensions for SMA.

BNC → BNC (*xval*)						
	MFS	k-NN-ORIG	k-NN-OMT		k-NN-SMA	
Dimensions (p)			25	200	25	200
TRAIN	52.0±1.3	62.7±1.4	64.5±1.2	**64.5**±1.4	-	-
TRAIN ∪ BNC	-	-	**69.7**±1.4	64.0±1.3	64.9±1.3	**64.7**±1.2
	MFS	SVM-ORIG	SVM-OMT		SVM-SMA	
Dimensions (p)			25	200	25	200
TRAIN	52.0±1.3	64.8±1.14	57.4±1.3	**61.5**±1.2	-	-
TRAIN ∪ BNC	-	-	**70.3**±1.3	65.6±1.1	66.0±1.4	**67.0**±1.1

Table V.1: Accuracy for the source scenario: training on labeled source corpus, plus unlabeled corpora.

V.3 Semi-supervised domain adaptation scenario

In this scenario we tried to adapt a general purpose supervised WSD system trained on the source corpus (BNC) to a target corpus (either SPORTS or FINANCE) using unlabeled corpora only. In addition we want to analyze the the effect of the source of unlabeled data.

The experiments were organized as follows. First, we trained each ML algorithm in each defined feature space (original,OMT, SMA) without any kind of unlabeled data. After that, we used unlabeled data in order to perform the adaptation. Thus, SVD features were induced from BNC (source domain) and SPORTS/FINANCE (target domain) and tested in the target domain.

V.3.1 Experimental results

Table V.2 shows the precision results for this scenario. Note that all methods have full coverage, i.e. they return a sense for all test examples, and therefore precision suffices to compare among systems. We have computed significance ranges for all results in this work using bootstrap resampling (Noreen, 1989). F_1 scores outside of these intervals are assumed to be significantly different from the related F_1 score ($p < 0.05$).

The table has two main parts, each regarding to one of the target domains, SPORTS and FINANCE. The use of two target domains allows to test whether the methods behave similarly in both domains. The columns denote the classifier and SVD method used: the MFS column corresponds to the most frequent sense, k-NN-ORIG (SVM-ORIG) corresponds to performing k-NN (SVM) on the original feature space, k-NN-OMT (SVM-OMT) corresponds to k-NN (SVM) on the reduced dimensions of the OMT strategy, and k-NN-SMA (SVM-SMA) corresponds to k-NN (SVM) on the reduced dimensions of the SMA strategy (cf. Section III.3). The rows correspond to the matrix used for SVD (cf. Section V.2.2). Note that some of the cells have no result, because that combination is not applicable, e.g. using the TRAIN ∪ BNC in the original space.

In the first row (TRAIN) of Table V.2 we can see that in both domains SVM on the original space outperforms k-NN with statistical significance. Those are the baseline systems. On the same row, working on the reduced space of the TRAIN matrix with OMT allows to improve the results of k-NN, but not for SVM.

Contrary to our expectations, adding target unlabeled corpora (TRAIN ∪ SPORTS and TRAIN ∪ FINANCE rows, respectively) does not improve the results over the baseline. But using the source unlabeled data (TRAIN ∪ BNC), we find that for both domains and in all four columns the results are significantly better than for the best baseline in both SPORTS and FINANCE corpora.

The best results on the TRAIN ∪ BNC row depend on the domain corpus. While k-NN-OMT obtains the best results for SPORTS, in FINANCE k-NN-SMA is best. k-NN, in principle a weaker method that SVM, is able to attain the same or superior performance than SVM on the reduced spaces.

Bnc → Sports				
matrix configuration	MFS	k-NN-ORIG	k-NN-OMT	k-NN-SMA
Train	39.0±1.3	51.7±1.3	53.0±1.6	-
Train ∪ Sports	-	-	47.8±1.5	49.7±1.5
Train ∪ Bnc	-	-	**61.4**±1.4	57.1±1.5
matrix configuration	MFS	SVM-ORIG	SVM-OMT	SVM-SMA
Train	39.0±1.3	53.9±1.3	47.4±1.5	-
Train ∪ Sports	-	-	51.8±1.5	53.8±1.5
Train ∪ Bnc	-	-	57.1±1.6	57.2±1.5
Bnc → Finance				
matrix configuration	MFS	k-NN-ORIG	k-NN-OMT	k-NN-SMA
Train	51.2±1.6	60.4±1.6	62.5±1.4	-
Train ∪ Finance	-	-	57.4±1.9	60.6±1.5
Train ∪ Bnc	-	-	65.9±1.5	**68.3**±1.4
matrix configuration	MFS	SVM-ORIG	SVM-OMT	SVM-SMA
Train	51.2±1.6	62.9±1.6	59.4±1.5	-
Train ∪ Finance	-	-	60.4±1.4	62.7±1.4
Train ∪ Bnc	-	-	67.0±1.3	66.8±1.5

Table V.2: Precision for the domain adaptation scenario: training on labeled source corpus, plus unlabeled corpora.

V.3.2 Controlling size

In the original experiments reported in the previous sections the size of the unlabeled corpora was not balanced. Due to the importance of the amount of unlabeled data, we performed two control experiments for the OMT and SMA matrices on the domain adaptation scenario, focusing on the k-NN method. Regarding OMT, we used the minimum number of instances per word between Bnc and each of the target domains. The system obtained 60.0 of precision using unlabeled data from Bnc and 49.5 for Sports data (compared to 61.4 and 47.8 in Table V.2, respectively). We did the same in the Finance domain, and we obtained 65.6 of precision for Bnc and 54.4 for Finance (compared to 65.7 and 57.4 in Table V.2, respectively). Although the contribution of Bnc unlabeled data is slightly lower in this experiment, due to the smaller amount of data, it still outperforms the target unlabeled data by a large margin. These results are shown in Table V.3.

In the case of the SMA matrix, we used 25% of the Bnc, which is comparable to the Sports and Finance sizes. The results, 56.9 of precision in Sports domain and 68.1 in Finance (compared to 57.1 and 68.3 in Table

Bnc → Sports	50% **corpus**	# Bnc = Sports
Bnc	61.4	60.0
Sports	47.8	49.5
Bnc → Finance	**50% corpus**	**# Bnc = Finance**
Bnc	65.7	65.6
Finance	57.4	54.4

Table V.3: Results of the experiments to control the unlabeled corpus size effect.

V.2, respectively), confirm that the size is not an important factor for SMA either.

V.4 Target scenario

In this second scenario we focus on the target domain. We train and test on the target domain, and use unlabeled data in order to improve the result. The goal of these experiments is to check the behavior of our method when applied to the target domain, in order to better understand the results on the domain adaptation scenario. They also provide an upperbound for semi-supervised domain adaptation.

V.4.1 Experimental results

The results are presented in table V.4. All experiments in this section have been performed using 3-fold cross-validation. Again, we have full coverage in all cases, and the significance ranges correspond to the 95% confidence level. The table has two main parts, each regarding to one of the target domains, Sports and Finance. As in Table V.2, the columns specify the classifier and SVD method used, and the rows correspond to the matrices used to obtain the features.

Table V.4 shows that k-NN-OMT using the target corpus (Sports and Finance, respectively) slightly improves over the k-NN-ORIG and SVM-ORIG classifiers, with significant difference in the Sports domain. Contrary to the results on the previous section, the source unlabeled corpus degrades performance, but the target corpus does allow for small improvements. Note that, in this scenario, both SVM and k-NN perform similarly in the original

SPORTS → SPORTS (*xval*)				
matrix configuration	MFS	*k*-NN-ORIG	*k*-NN-OMT	*k*-NN-SMA
TRAIN	77.8±1.2	84.5±1.0	85.0±1.1	-
TRAIN ∪ SPORTS	-	-	**86.1**±0.9	82.7±1.1
TRAIN ∪ BNC	-	-	84.4±1.0	80.4±1.5
matrix configuration	MFS	SVM-ORIG	SVM-OMT	SVM-SMA
TRAIN	77.8±1.2	85.1±1.0	81.0±1.5	-
TRAIN ∪ SPORTS	-	-	85.1±1.1	80.3±1.5
TRAIN ∪ BNC	-	-	84.3±0.9	79.8±1.2
FINANCE → FINANCE (*xval*)				
matrix configuration	MFS	*k*-NN-ORIG	*k*-NN-OMT	*k*-NN-SMA
TRAIN	82.3±1.3	87.1±1.0	87.4±1.0	-
TRAIN ∪ SPORTS	-	-	**87.8**±0.8	84.3±1.4
TRAIN ∪ BNC	-	-	87.4±1.2	83.5±1.2
matrix configuration	MFS	SVM-ORIG	SVM-OMT	SVM-SMA
TRAIN	82.3±1.3	87.0±1.0	85.5±1.1	-
TRAIN ∪ SPORTS	-	-	86.4±0.9	82.9±1.1
TRAIN ∪ BNC	-	-	85.7±0.9	84.3±1.1

Table V.4: Precision for the target scenario: training on labeled target corpora, plus unlabeled corpora.

space, but only *k*-NN is able to profit from the reduced space. Table V.5 summarizes the best result, alongside the error reduction.

The results of these experiments allow to contrast both scenarios, and to get deeper insight about the relation between the labeled and unlabeled data when performing SVD, as we will examine in the next section.

V.5 Discussion

The main contribution of this dissertation is to show that we obtain robustness when faced with domain shifts using a semi-supervised strategy. We show that we can obtain it using a large, general, unlabeled corpus. Note that our semi-supervised method to attain robustness for domain shifts is very cost-effective, as it does not require costly hand-tagged material nor even large numbers of unlabeled data from each target domain. These results are more valuable given the lack of substantial positive results on the literature on semi-supervised or supervised domain adaptation for WSD (Escudero *et al.*, 2000; Martínez and Agirre, 2000; Chan and Ng, 2007).

Compared to other settings, our semi-supervised results improve over the completely unsupervised system in (Koeling *et al.*, 2005), which had 43.7% and 49.9% precision for the SPORTS and FINANCE domains respectively, but lag well behind the target domain scenario, showing that there is still room for improvement in the semi-supervised setting.

While these results are based on a lexical sample, and thus not directly generalizable to an all-words corpus, we think that they reflect the main trends for nouns, as the 41 nouns where selected among those exhibiting domain dependence (Koeling *et al.*, 2005). We can assume, though it would be needed to be explored empirically, that other nouns exhibiting domain independence would degrade less when moving to other domains, and thus corroborate the robustness effect we have discovered.

Table V.5 summarizes the main results, and also shows the error reduction figures, which range between 6.9% and 16.3%. As the most important conclusion, we want to stress that, in this scenario, we are able to build a very robust system just adding unlabeled source material, and that we fail to adapt to the domain using the target corpus. These results are relevant to improve a generic WSD system to be more robust when ported to new domains.

The fact that we attain **robustness** rather than domain adaptation proper deserves some analysis. In the domain adaptation scenario only source unlabeled data helped, but the results on the target scenario show that it is the target unlabeled data which is helping, and not the source one. Given that SVD basically finds correlations among features, it seems that constructing the term-by-document (or feature-by-example) matrix with the training data and the unlabeled corpus related to the training data is the key factor in play here.

The reasons for this can be traced back as follows. Our source corpus is the BNC, which is a balanced corpus containing a variety of genres and domains. The 100 examples for each word that have been hand-tagged were gathered at random, and thus cover several domains. For instance, the OMT strategy for building the matrix extracts hundreds of other examples from the BNC, and when SVD collapses the features into a reduced space, it effectively captures the most important correlations in the feature-by-example matrix. When faced with examples from a new domain, the reduced matrix is able to map some of the features found in the test example to those in the train example. Such overlap is more difficult if only 100 examples from the source domain are available. The unlabeled data and SVD process allow to capture correlations among the features occurring in the test data and those in the training data.

On the other hand, we are discarding all original features, as we focus on the features from the reduced space alone. The newly found correlations come at the price of possibly ignoring effective original features, causing information loss. Only when the correlations found in the reduced space outweigh this information loss do we get better performance on the reduced space than in the original space. The experiment in Section V.4 is important in that it shows that the improvement is much smaller and only significant in the target domain scenario, which is in accordance with the hypothesis above. This information loss is a motivation for the combination of the features from the reduced space with the original features, which will be the focus on next chapter.

Regarding the learning method and the two strategies to apply SVD, the results show that k-NN profits from the reduced spaces more than SVM, even if its baseline performance is lower than SVM. Regarding the matrix building system, in the domain adaptation scenario, k-NN-OMT obtains the best results (with statistical significance) in the SPORTS corpus, and k-NN-SMA yields the best results (with statistical significance) in the FINANCE domain. Averaging over both domains, k-NN-OMT is best. The target scenario results confirm this trend, as k-NN-OMT is superior to k-NN-SMA in both domains. These results are in accordance with our previous experience in Chapter IV, where our OMT method got better results than SMA and those of Gliozzo *et al.* (2005) (who also use a method similar to SMA) on the Senseval-3 lexical sample. While OMT reduces the feature-by-example matrix of each target word, SMA reduces a single term-by-document matrix. SMA is able to find important correlations among similar terms in the corpus, but it misses the rich feature set used by WSD systems, as it focuses on bag-of-words alone. OMT on the other hand is able to find correlations between all features which are relevant to the target word only.

V.6 Conclusions

In this chapter we explore robustness and domain adaptation issues for Word Sense Disambiguation using SVD and unlabeled data. We focus on the semi-supervised scenario, where we train on the source corpus (BNC), test on two target corpora (SPORTS and FINANCE sections of Reuters), and improve the results using unlabeled data.

Our method yields up to 16.3% error reduction compared to SVM and k-NN on the labeled data alone, showing the first positive results on domain adaptation for WSD. In fact, we show that our results are due to the use of a large, general, unlabeled corpus, and rather than domain-adaptation proper we show robustness in face of a domain shift. This kind of robustness is even more cost-effective than semi-supervised domain adaptation, as it does not require large unlabeled corpora or repeating the computations for each new target domain.

These experiments show that the OMT technique to apply SVD that we proposed in Section III.3.3.2 compares favorably to SMA, which has been previously used in (Gliozzo *et al.*, 2005), and that k-NN excels SVM on the features from the reduced space. We also show that the unlabeled data needs to be related to the training data, and that the benefits of our method are larger when faced with a domain shift (compared to test data coming from the same domain as the training data).

In the next chapters we combine the features from the reduced space with the rest of the features, either using a combination of k-NN classifiers (cf. Section III.4, and Chapter IV for experimental results) or a more complex kernel combination (cf. Section III.4), where it is natural extension to apply this techniques to the supervised domain adaptation scenario. This way, we pretend to take advantage of both knowledge representation and try to minimize the information loss from both spaces.

SPORTS	FINANCE	sign.	E.R (%)	method
53.9±1.3	62.9±1.6	-	-	labeled source (SVM-ORIG: baseline)
57.1±1.5	**68.3±1.4**	++	6.9/14.5	**labeled source** + SVD **on unlabeled source** (*k*-NN-SMA)
61.4±1.4	65.9±1.5	++	16.3/8.1	**labeled source** + SVD **on unlabeled source** (*k*-NN-OMT)
85.1±1.0	87.0±1.0	-	-	labeled target (SVM-ORIG: baseline)
86.1±0.9	**87.8±0.8**	+	6.7/6.1	**labeled target** + SVD **on unlabeled target** (*k*-NN-OMT)

Table V.5: Summary with the most important results for the two scenarios (best results for each in bold). The significance column shows significance over baselines: ++ (significant in both target domains), + (significant in a single domain). The E.R column shows the error reduction in percentages over the baseline methods.

CHAPTER VI

Supervised domain adaptation

We have shown that SVD *in combination with unlabeled data is a reliable way to find better correlations among features. By mixing both source and target domains in the training set, we hypothesized that*SVD *would help to find a bridge between the features in source and target domains. Our results show that it is possible to build a feature space where the gap between domains is smaller, and that the examples from the source general domain are useful, and lead to the first positive supervised domain adaptation results to date. In this chapter we will show that our* WSD *system trained on a general source corpus (* BNC*) and the target corpus obtains up to 22% error reduction when compared to a system trained on the target corpus alone. In addition, we show that as little as 40% of the target corpus (when supplemented with the source corpus) is sufficient to obtain the same results as training on the full target data. The key for success is the use of unlabeled data with* SVD*, a combination of kernels and* SVM*.*

VI.1 Introduction

In Chapter V we have shown that ML models for WSD (for NLP in general) suffer from the domain shift problem. Their perfomance drop drastically when training data and testing data are comming from other domains.

In this chapter we will focus on supervised WSD adaptation. Following the work done in the previous chapter, we will compare the performance of similar supervised WSD systems on three different scenarios: In the **source scenario** the WSD system is trained on the source domain and tested on the target domain (this scenario is comparable to the one mentioned in the previous chapter). In the **target scenario** the WSD system is trained and tested on the target domain (using cross-validation). In the **supervised domain adaptation scenario** the WSD system is trained on both source and target domain and tested in the target domain (also using cross-validation over the target data). Note that we have already studied the source to target and the target scenarios in Chapter V.

The source scenario represents a weak baseline for domain adaptation, as it does not use any examples from the target domain. The target scenario represents the hard baseline, and in fact, if the domain adaptation scenario does not yield better results, the adaptation would have failed, as it would mean that the source examples are not useful when we do have hand-labeled target examples.

Previous work shows that current state-of-the-art WSD systems are not able to obtain better results on the adaptation scenario compared to the target scenario (Escudero *et al.*, 2000; Agirre and Martínez, 2004b; Chan and Ng, 2007). This would mean that if a user of a generic WSD system (i.e. based on hand-annotated examples from a generic corpus) would need to adapt it to a specific domain, he would be better off throwing away the generic examples and hand-tagging domain examples directly. This chapter will show that domain adaptation is feasible, even for difficult domain-related words, in the sense that generic corpora can be reused when deploying WSD systems in specific domains. We will also show that, given the source corpus, our technique can save up to 60% of effort when tagging domain-related occurrences.

In this chapter we will also present additional experiments, such as learning curves, where we will study the effect of different numbers of target tagged data added into the training set. The purposed method obtains good adaptation with little target data adding to source training set. We used the method purposed by Daumé III (2007) to have a reference and deeper conclusions. We will show that while the latter method can outperform the simplest baselines, and get somehow adapted, is not able to do better than the stronger baselines.

The chapter is structured as follows. Section IV.2.1 reviews datasets and the learning algorithms, including system combination. The experimental results are presented in Section VI.3, Section VI.4 and Section VI.5, for source scenario, target scenario and supervised domain adaptation scenario, respectively. Section VI.6 presents the discussion and some analysis of this chapter and finally, Section VI.7 draws the conclusions.

VI.2 Experimental settings

Regarding the experiments, we have used the identical experimental settings we used in the previous chapter[1]. In short, the dataset (Koeling *et al.*, 2005) consists of the examples coming from the BNC (Leech, 1992) and the SPORTS and FINANCE sections of the Reuters corpus (Rose *et al.*, 2002), comprising around 300 examples (roughly 100 from each of those corpora) for each of the 41 nouns. The nouns were selected because they were salient in either the SPORTS or FINANCE domains, or because they had senses linked to those domains. The occurrences were hand-tagged with the senses from WordNet (WN) version 1.7.1 (Fellbaum, 1998). The unlabeled data also is coming from BNC, SPORTS and FINANCE. It is important to note that in these experiments we always have used unlabeled data related to the training data. Thus, if the training set consist of the BNC and SPORTS, we use mixed unlabeled data from the BNC and SPORTS, in equal quantity of examples for each domain.

Concerning learning algorithms, again, we test an k-NN algorithm and a linear SVM classifier. In k-NN the cosine was used and the degree of neighbors (k) was set in 5. For SVM, the soft margin parameter is highly dependent on the feature set and training dataset, and given that we will be doing experiments combining different examples, we opted to avoid overfitting C. For the kernelized version (see below) we used the default C value, and for the linear we have utilized the same values as in Chapter V. Note that all methods presented here have full coverage, i.e. they return a sense for all test examples, and therefore precision equals recall, and suffices to compare among systems.

With respect to learning features, we used the so-called original features, which consist of local collocations features, syntactic features, and bag-of-words features (additional details in Section III.1) ; and the SVD features:

[1]Note that although they are reported separately, both chapters represent a continuum

OMT and SMA features (further details in Section III.3.2.1 and Section V.2.2).

Combination of feature spaces

In short, we have used the following combination techniques in this chapter:

k-NN **combination** (more details in Section III.4): For this experiments we built three classifiers trained on OMT, SMA and original features. In order to combine them we weight each vote (one of the k neighbors for each classifier) by the inverse ratio of its position in the rank of the single classifier.

Kernel combination (further details in Section III.4): the combined kernel is a combination of a normalized linear kernel, an OMT kernel and an SMA kernel. Due to the domain differences, we did not optimize the C parameter.

Feature augmentation method (cf. Section III.2.1.6): the main idea is to give more importance to those examples coming from the same domain. As we are testing in the target domain, we would prefer to give more discriminative power to the instances from training which are from the target domain. Essentially, all we did was to make three version of the original feature vector: a general version, a source-specific version and a target-version. The augmented source data contained only general and source-specific versions and the augmented target data contains general and target-specific versions. Thus, the target data has twice influence than source when making prediction about test target data.

VI.3 Source scenario

In this scenario our supervised WSD systems are trained on the general source corpus (BNC) and tested on the specific target domains separately (SPORTS and FINANCE). We do not perform any kind of adaptation, and therefore the results are those expected for a generic WSD system when applied to domain-specific texts.

Table VI.1 shows the results for k-NN and SVM trained with the original features on the BNC. In addition, we also show the results for the Most Frequent Sense baseline (MFS) taken from the BNC. The second column denotes the accuracies obtained when testing on SPORTS, and the third col-

BNC → $\mathcal{X}$	SPORTS	FINANCE
MFS	39.0	51.2
k-NN	51.7	60.4
SVM	**53.9**	**62.9**

Table VI.1: Source to target results: Train on BNC, test on SPORTS and FINANCE.

	SPORTS		FINANCE	
$\mathcal{X} \rightarrow \mathcal{X}$	TRAIN	+UNLAB	TRAIN	+UNLAB
MFS	77.8	-	82.3	-
k-NN	84.5	-	87.1	-
SVM	85.1	-	87.0	-
k-NN-OMT	85.0	86.1	87.3	87.6
SVM-OMT	82.9	85.1	85.3	86.4
k-NN-SMA	-	81.1	-	83.2
SVM-SMA	-	81.3	-	84.1
k-NN-COMB	86. 0	**86.7**	87.9	**88.6**
SVM-COMB	-	86.5	-	88.5

Table VI.2: Target results: train and test on SPORTS, train and test on FINANCE, using 3-fold cross-validation.

umn the accuracies for FINANCE. The low accuracy obtained with MFS, e.g. 39.0 of precision in SPORTS, shows the difficulty of this task. Both classifiers improve over MFS. These classifiers are weak baselines for the domain adaptation system.

VI.4 Target scenario

In this scenario we lay the harder baseline which the domain adaptation experiments should improve on (cf. next section). The WSD systems are trained and tested on each of the target corpora (SPORTS and FINANCE) using 3-fold cross-validation.

Table VI.2 summarizes the results for this scenario. TRAIN denotes that only tagged data was used to train, +UNLAB denotes that we added unlabeled data related to the source corpus when computing SVD. The rows denote the classifier and the feature spaces used, which are organized in four sections. On the top rows we show the three baseline classifiers on the original features. The two sections below show the results of those classifiers on the reduced dimensions, OMT and SMA. Finally, the last rows show the results of the combination strategies. Note that some of the cells have no result, because that combination is not applicable (e.g. using the train and unlabeled data in the original space).

First of all note that the results for the baselines (MFS, SVM, k-NN) are much larger than those in Table VI.1, showing that this dataset is specially demanding for supervised WSD, and particularly difficult for domain adaptation experiments. These results seem to indicate that the examples from the source general corpus could be of little use when tagging the target corpora. Note specially the difference in MFS performance. The priors of the senses are very different in the source and target corpora, which is a well-known shortcoming for supervised systems. Note the high results of the baseline classifiers, which leave small room for improvement.

The results for the more sophisticated methods show that SVD and unlabeled data helps slightly, except for k-NN-OMT on Sports. SMA decreases the performance compared to the classifiers trained on original features. The best improvements come when the three strategies are combined in one, as both the kernel and k-NN combinations obtain improvements over the respective single classifiers. Note that both the k-NN and SVM combinations perform similarly.

In the combination strategy we show that unlabeled data helps slightly, because instead of only combining OMT and original features we have the opportunity to introduce SMA. Note that it was not our aim to improve the results of the basic classifiers on this scenario, but given the fact that we are going to apply all these techniques in the domain adaptation scenario, we need to show these results as baselines. That is, in the next section we will try to obtain results which improve significantly over the best results in this section.

VI.5 Supervised Domain adaptation scenario

In this last scenario we try to show that our WSD system trained on both source (Bnc) and target (Sports and Finance) data performs better than the one trained on the target data alone. We also use 3-fold cross-validation

	SPORTS		FINANCE	
BNC + $\mathcal{X} \rightarrow \mathcal{X}$	TRAIN	+ UNLAB	TRAIN	+ UNLAB
BNC $\rightarrow \mathcal{X}$	53.9	-	62.9	-
$\mathcal{X} \rightarrow \mathcal{X}$	86.0	86.7	87.9	88.5
MFS	68.2	-	73.1	-
k-NN	81.3	-	86.0	-
SVM	84.7	-	87.5	-
k-NN-OMT	84.0	84.7	87.5	86.0
SVM-OMT	85.1	84.7	84.2	85.5
k-NN-SMA	-	77.1	-	81.6
SVM-SMA	-	78.1	-	80.7
k-NN-COMB	84.5	87.2	88.1	88.7
SVM-COMB	-	**88.4**	-	**89.7**
SVM-AUG	85.9	-	88.1	-

Table VI.3: Domain adaptation results: Train on BNC and SPORTS, test on SPORTS (same for FINANCE).

for the target data, but the entire source data is used in each turn. The unlabeled data here refers to the combination of unlabeled source and target data.

The results are presented in table VI.3. Again, the columns denote if unlabeled data has been used in the learning process. The rows correspond to classifiers and the feature spaces involved. The first rows report the best results in the previous scenarios: BNC $\rightarrow \mathcal{X}$ for the source to target scenario, and $\mathcal{X} \rightarrow \mathcal{X}$ for the target scenario. The rest of the table corresponds to the domain adaptation scenario. The rows below correspond to MFS and the baseline classifiers, followed by the OMT and SMA results, and the combination results. The last row shows the results for the feature augmentation algorithm (Daumé III, 2007).

Focusing on the results, the table shows that MFS decreases with respect to the target scenario (cf. Table VI.2) when the source data is added, probably caused by the different sense distributions in BNC and the target corpora. The baseline classifiers (k-NN and SVM) are not able to improve over the baseline classifiers on the target data alone, which is coherent with past research, and shows that straightforward domain adaptation does not work.

The following rows show that our reduction methods on themselves (OMT,

	Sports	Finance	significance test		
			svm (t)	k-nn-comb	svm-aug
Bnc $\rightarrow \mathcal{X}$					
mfs	39.0	51.2			
svm	53.9	62.9			
$\mathcal{X} \rightarrow \mathcal{X}$					
mfs	77.8	82.3			
svm	85.1	87.0			
k-nn-comb (+unlab)	86.7	88.6			
Bnc $+\mathcal{X} \rightarrow \mathcal{X}$					
mfs	68.2	73.1			
svm	84.7	87.5			
svm-aug	85.9	88.1	++	- -	-
svm-comb (+unlab)	**88.4**	**89.7**	++	++	++

Table VI.4: The most important results in each scenario. The significance columns shows the significance over the baselines and svn-aug: ++ (significant in both domains), - - (not significant in both domain). svm(t) denote the svm baseline in target scenario.

sma used by k-nn and svm) also fail to perform better than in the target scenario, but the combinations using unlabeled data (k-nn-comb and specially svm-comb) do manage to improve the best results for the target scenario, showing that we were able to attain domain adaptation. The feature augmentation approach (svm-aug) does improve slightly over svm in the target scenario, but not over the best results in the target scenario, showing the difficulty of domain adaptation for wsd, at least on this dataset.

VI.6 Discussion and analysis

Table VI.4 summarizes the most important results. The kernel combination method with unlabeled data on the adaptation scenario reduces the error on 22.1% and 17.6% over the baseline svm on the target scenario (Sports and Finance respectively), and 12.7% and 9.0% over the k-nn combination method on the target scenario. These gains are remarkable given the already high baseline, specially taking into consideration that the 41 nouns are closely related to the domains. The differences, including svm-aug and k-nn-comb, are statistically significant according to the Wilcoxon rank sum test with

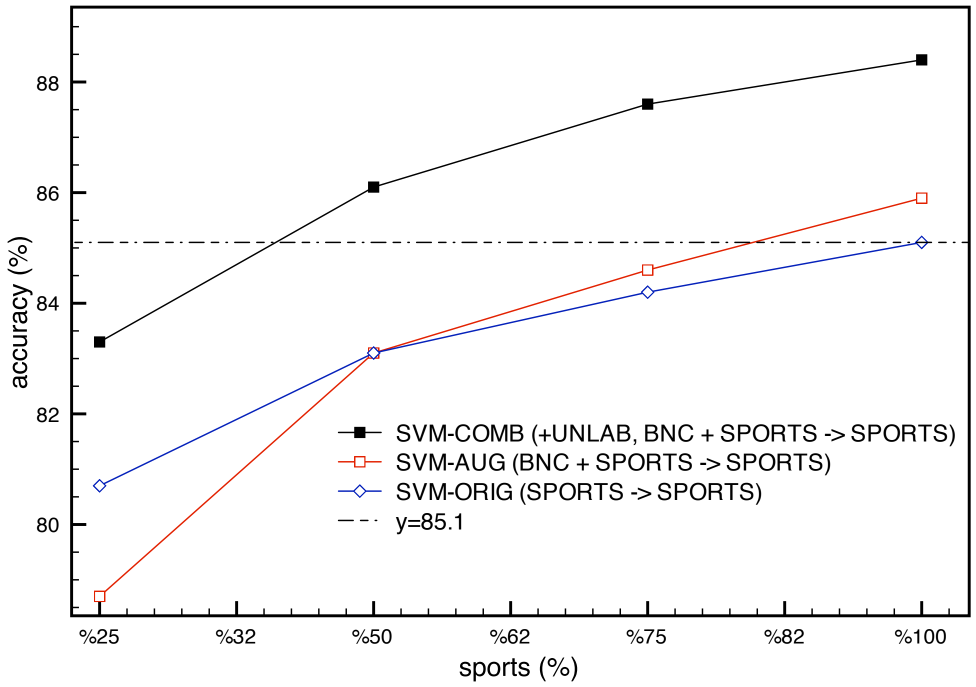

Figure VI.1: Learning curves for SPORTS. The $\mathcal{X}$ axis denotes the amount of SPORTS data and the Y axis corresponds to accuracy.

$p < 0.01$.

In addition, we carried extra experiments to examine the learning curves, and to check, given the source examples, how many additional examples from the target corpus are needed to obtain the same results as in the target scenario using all available examples. We fixed the source data and used increasing amounts of target data. We show the original SVM on the target scenario, and SVM-COMB (+UNLAB) and SVM-AUG as the domain adaptation approaches. The results are shown in Figure VI.1 for SPORTS and Figure VI.2 for FINANCE. The horizontal line corresponds to the performance of SVM on the target domain. The point where the learning curves cross the horizontal line show that our domain adaptation method needs only around 40% of the target data in order to get the same performance as the baseline SVM on the target data. The learning curves also shows that the domain adaptation kernel combination approach, no matter the amount of target data, is always above the rest of the classifiers, showing the robustness of our approach.

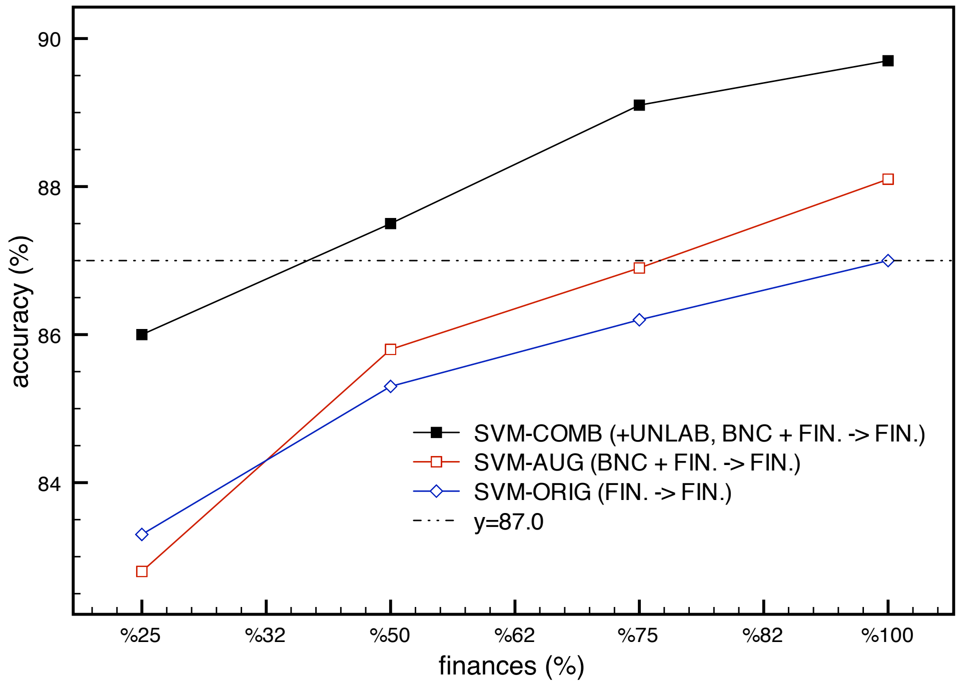

Figure VI.2: Learning curves for FINANCE. The X axis denotes the amount of FINANCE data and Y axis corresponds to the accuracy.

VI.7 Conclusion and future work

In this chapter we explored supervised domain adaptation for WSD with positive results, showing that hand-labels from the general domain (source text) are useful when training a WSD system that is be applied to specific domains (target texts). We performed several experiments in three scenarios. In the first scenario (source scenario), the classifiers were trained on general source domain data (the BNC) and tested on the target domains, composed by the SPORTS and FINANCE sections of Reuters. In the second scenario (target scenario) we set the main baseline for our domain adaptation experiment, training and testing our classifiers on the target domain data. In the last scenario (domain adaptation scenario), we combined both source and target data for training, and test on the target data.

We reported results in each scenario for k-NN and SVM classifiers, for reduced features obtained using SVD over the training data, for the use of unlabeled data, and for k-NN and SVM combinations of all.

Our results show that our best domain adaptation strategy (using kernel combination of SVD features and unlabeled data related to the training data) yields statistically significant improvements: up to 22% error reduction compared to SVM on the target domain data alone. We also show that our domain adaptation method only needs 40% of the target data (in addition to the source data) in order to get the same results as SVM on the target alone.

We obtain coherent results in two target scenarios, and consistent improvement at all levels of the learning curves, showing the robustness of our findings. We think that our dataset, which comprises examples for 41 nouns that are closely related to the target domains, is specially demanding, as one would expect the performance of a generic WSD system to drop when moving to the domain corpus, **specially** for domain-related words, while we could expect the performance to be similar for generic or unrelated words.

In terms of contributions we may summarize the chapter as follows:

- First successful results in supervised domain adaptation in the supervised scenario
- Our WSD system was able to take profit from the source general domain tagged examples.
- Kernel methods are a good option to combine different feature spaces in a simple and elegant way.

For the future, it would be interesting to evaluate our method on all words datasets (e.g. DSO or OntoNotes), to confirm whether our the positive results are confirmed[2]. We would also like to study word-by-word behavior, in order to assess whether target examples are really necessary for words which are less related to the domain.

[2]We are organizing domain-specific WSD task for SemEval-2010: `http://xmlgroup.iit.cnr.it/SemEval2010/`

CHAPTER VII

Knowledge-based domain-specific WSD

This chapter explores an alternative method for WSD *of specific-domains. We used a state-of-the-art graph based* WSD *system (introduced by Agirre and Soroa (2009)) that uses the information in WordNet. Evaluation was performed in the same framework as the two previous chapters. The results show that in all three corpora the knowledge-based* WSD *algorithm improves over previous knowledge-based results, and also over two state-of-the-art supervised* WSD *systems trained on SemCor, the largest publicly available annotated corpus. We also show that disambiguating automatically built thesauri (instead of the actual occurrence contexts) yields better results on the domain datasets, but not on the general one. Interestingly, the results are higher for domain-specific corpus than for the general corpus, raising prospects for improving current* WSD *systems when applied to specific domains.*

VII.1 Introduction

The state-of-the-art in WSD has shown the best performing systems are those based on supervised learning (cf. Section II.4). Despite the impressive results, they need large amounts of hand-tagged data to deal with data sparseness and domain shift problems. The sparseness and domain shift problems are specially acute when deploying a supervised WSD system on a specific domain. Hand tagging examples for every new domain would provide the desired performance, but it is unfeasible in practice, because of the high manual cost involved. In Chapter V and Chapter VI we have showed that when the training and test data come from different domain the performance decrease significantly, and presented a method to alleviate those problems.

As an alternative to supervised systems, knowledge-based WSD is re-emerging as a powerful alternative. Knowledge-based systems exploit the information in a Lexical Knowledge Base (LKB) to perform WSD, without using any corpus evidence. In particular, graph-based methods are getting increasing attention from the WSD community (Sinha and Mihalcea, 2007; Navigli and Lapata, 2007). These methods use well-known graph-based techniques to find and exploit the structural properties of the graph underlying a particular LKB. In (Agirre and Soroa, 2009) authors proposed a graph-based algorithm using Personalized PageRank which outperformed other unsupervised WSD systems in publicly available datasets. In this chapter we explore the application of their algorithm to domain-specific corpora.

Our work here focuses on the comparison between state-of-the-art supervised and knowledge-based WSD systems on specific domains, and to study better ways to apply knowledge-based WSD methods on specific domains. Our proposal can also be seen as a continuation of (Koeling *et al.*, 2005), and we show that our WordNet-based WSD method yields better results. We also study whether the strategy to select one predominant sense for the whole corpus using the thesaurus performs better than disambiguating each occurrence of the word separately. This chapter is complementary to those experiments performed with the supervised methods in previous chapters, and holds promise for potential combinations.

The chapter is structured as follows. Section VII.2 briefly reviews the supervised systems used for comparison purposes. Section VII.3 presents the graph-based techniques, which are applied to WSD. In Section VII.4 the evaluation framework and results are presented. Finally, the conclusions are drawn and further work is mentioned.

VII.2 Supervised WSD

As baselines, we use two state-of-the-art WSD classifiers: Support Vector Machines (SVM) and k-Nearest Neighbors (k-NN), respectively.

Regarding k-NN, the similarity among instances was measured as the cosine of their featured vectors. We set k to 5 based on previous work. Regarding SVM, we used linear kernels, due to the high amount of learning features. No soft margin (C) was estimated for any baseline system and the default C was used. We used the one-versus-all strategy, as implemented in SVM-Light.

In order to train the classifiers, we relied on the original features described in Section III.1. The feature space consist of local collocation, syntactic dependencies, and bag-of-words. Both systems were trained on SemCor, which we mapped it from WordNet 1.6 to WordNet 1.7.1 (Daude *et al.*, 2000). In the case where target word has fewer than 10 instances in SemCor we have applied the most frequent sense, as customary in all-words supervised WSD systems. For the 41 words in the evaluation dataset (cf. Section VII.4) 8 words had less than 10 training instances. The maximum amount of training instances was 114, with an average of 37.

VII.3 Lexical Knowledge based WSD

In this section we will briefly explain how to apply PageRank and Personalized PageRank to knowledge-based WSD, as introduced in (Agirre and Soroa, 2009). We give further details in Section II.3.1.2.

A Lexical Knowledge Base (LKB) is formed by a set of concepts and relations among them, plus a dictionary, i.e. a list of words (typically, word lemmas) each of them linked to at least one concept of the LKB. Such a LKB can be naturally represented as an undirected graph $G = (V, E)$ where nodes represent LKB concepts (v_i), and each relation between concepts v_i and v_j is represented by an undirected edge $e_{i,j}$. In this work, we used WordNet 1.7 as the LKB, using all relations supplemented with disambiguated glosses as provided by the Extended WordNet. This setting was optimal in (Agirre and Soroa, 2009). The WordNet version follows that of the evaluation dataset (cf. Section VII.4).

VII.3.1 Static PageRank (PR), no context

If we apply traditional PageRank over the whole WordNet, we get a context-independent ranking of word senses. All concepts in WordNet get ranked according to their PageRank value. Given a target word, it suffices to check which is the relative ranking of its senses, and the WSD system would output the one ranking highest. We call this application of PageRank to WSD *Static PageRank*, as it does not change with the context, and we use it as a baseline.

As PageRank over undirected graphs is closely related to the degree, the Static PageRank returns the most predominant sense according to the number of relations the senses have. We think that this is closely related to the Most Frequent Sense attested in general corpora, as the lexicon builders would tend to assign more relations to the most predominant sense. In fact, our results (cf. Section VII.4.1) show that this is indeed the case, at least for the English WordNet.

VII.3.2 Personalized PageRank (PPR) using context

Static PageRank is independent of context, but this is not what we want in a WSD system. Given an input piece of text we want to disambiguate all content words in the input according to the relationships among them. For this we can use Personalized PageRank over the whole WordNet graph.

Given an input text (a sentence in our case), we extract the list W_i $i = 1 \ldots m$ of content words (i.e. nouns, verbs, adjectives and adverbs) which have an entry in the dictionary, and thus can be related to LKB concepts. Note that monosemous words will be related to just one concept, whereas polysemous words may be attached to several. As a result of the disambiguation process, every LKB concept receives a score. Then, for each target word to be disambiguated, we just choose its associated concept in G with maximal score.

In order to apply *Personalized PageRank* over the LKB graph, the context words are first inserted into the graph G as nodes, and linked with directed edges to their respective concepts. Then, the Personalized PageRank of the graph G is computed by concentrating the initial probability mass uniformly over the newly introduced word nodes. As the words are linked to the concepts by directed edges, they act as source nodes injecting mass into the concepts they are associated with, which thus become relevant nodes, and spread their mass over the LKB graph. Therefore, the resulting Personalized PageRank vector can be seen as a measure of the structural relevance of LKB concepts in the presence of the input context.

This method has one problem: if one of the target words has two senses which are related to each other by semantic relations, those senses would reinforce each other, and could thus dampen the effect of the other senses in the context. With this observation in mind authors have used a variant where, for each target word W_i, the initial probability mass is concentrated in the senses of the words surrounding W_i, but not in the senses of the target word itself, avoiding to bias the initial score of concepts associated to target word W_i. Agirre and Soroa (2009) show that this variant gets the best results.

Given the fact that finding out the predominant sense seems a powerful option, we decided to try two further variants of the Personalized PageRank WSD algorithm. Instead of returning a different sense for each occurrence, we also evaluated the results of selecting the sense which is chosen most frequently by Personalized PageRank for the target word (*PPRank.maxsense* variant). Another alternative is to join all contexts of the target word into a single large context and then disambiguate the target word using this large context in a single run (*PPRank.all-in-one* variant).

VII.3.3 Personalized PageRank (PPR) using related words

Instead of disambiguating the target word using the occurrence context, we could follow (Koeling *et al.*, 2005) and disambiguate the target word using the set of related words as collected from the target corpus (cf. Section II.1.2). We would thus annotate all the occurrences of the target word in the test corpus with the same sense. For instance, in the SPORTS corpus, instead of disambiguating the word *coach* using each of its occurrences as context (e.g. "*Has never won a league title as a coach but took Parma to success in Europe ...*"), we would disambiguate *coach* using its most related words according to the thesaurus (e.g. *manager, captain, player, team, striker, ...*). In this work we use the automatically constructed thesauri built by Koeling *et al.* (2005), one for each corpus of the evaluation dataset, i.e. SPORTS, FINANCE and general. Given a target noun w, Koeling *et al.* obtained a set of co-occurrence triples $\langle w, r, x \rangle$ and associated frequencies, where r is a grammatical relation and x the co-occurring word in that relation. For every pair of nouns, they computed their distributional similarity comparing their respective triples using the measure suggested by Lin (1998). Finally, the 50 most similar nouns are retrieved for each target noun.

Systems		BNC	SPORTS	FINANCE
Baselines	Random	*19.7	*19.2	*19.5
	SemCor MFS	*34.9 [33.60, 36.20]	*19.6 [18.40, 20.70]	*37.1 [35.70, 38.00]
	Static PRank	*36.6 [35.30, 38.00]	*20.1 [18.90, 21.30]	*39.6 [38.40, 41.00]
Supervised	SVM	*38.7 [37.30, 39.90]	*25.3 [24.00, 26.30]	*38.7 [37.10, 40.10]
	k-NN	42.8 [41.30, 44.10]	*30.3 [29.00, 31.20]	*43.4 [42.00, 44.80]
Context	PPRank	**43.8** [42.40, 44.90]	*35.6 [34.30, 37.00]	*46.9 [45.39, 48.10]
	PPRank.maxsense	*39.3 [38.00, 40.60]	*36.0 [34.70, 37.40]	*53.1 [51.70, 54.40]
	PPRank.all-in-one	*39.6 [38.20, 40.90]	*42.5 [41.20, 43.90]	*46.4 [44.90, 47.80]
Related words	Koeling *et al.* (2005)	*40.7 [39.20, 42.00]	*43.3 [42.00, 44.60]	*49.7 [48.00, 51.10]
	PPRank	*37.7 [36.30, 39.00]	**51.5** [50.00, 52.90]	**59.3** [57.80, 60.70]
	PPRank.th+ctx	*38.2 [36.70, 39.50]	49.9 [48.50, 51.60]	57.8 [56.40, 59.20]
Upperbound	Test MFS	*52.0 [50.60, 53.30]	*77.8 [76.60, 79.00]	*82.3 [81.00, 83.30]

Table VII.1: Recall of baselines and systems on each of the corpus (SPORTS, FINANCE and BNC), including confidence intervals. * means statistically significant compared to the best system in each column (in bold). *Context* rows correspond to Personalized PageRank using occurrence context (cf. Section VII.3.2). *Related words* rows correspond to systems using related words as context (cf. Section VII.3.3).

VII.4 Evaluation Framework and results

We used the evaluation dataset published in Koeling *et al.* (2005), which consists of examples from the Sports and Finance sections of the Reuters corpus, and Bnc. The selected 41 words are quite polysemous and thus difficult to disambiguate, with an average polysemy of 6.7 senses, ranging from 2 to 13 senses.

(Koeling *et al.*, 2005) did not clarify how did they select the "correct" sense, and we decided to choose the sense selected by the majority of taggers. In case of ties we discarded the occurrence from the test set. This, and the fact that Koeling *et al.* discarded one of the target words, make our figures slightly different from those in (Koeling *et al.*, 2005).

VII.4.1 Experimental results

As evaluation measure we use recall, the number of correct occurrences divided by the total number of occurrences. Recall is more informative than accuracy, as some methods failed to return results for a handful of occurrences. Table VII.1 shows the results of the different WSD approaches on the different corpora (expressed in three main columns). The confidence interval is also shown, as computed using bootstrap resampling with 95% confidence.The systems in the table are divided in four groups.

The first rows report the baseline approaches, such as the random baseline, the most frequent sense as attested in SemCor and the results of the static PageRank. In the second group of rows, the results for supervised systems, k-NN and SVM, are shown. Next, we show the Personalized PageRank over occurrence context in its three variants (cf. Section VII.3.2). Below we show the approaches based on related words, including the results of Koeling *et al.* and the combination of applying our algorithm to the related words and each context (*th+ctx*). Finally, we show the most frequent sense according to the test data, which can be seen as an upperbound of our algorithm. We will now consider several issues in turn.

Baselines and supervised systems: The results show that SemCor MFS is very low, close to the random baseline and far from the Test MFS, especially for the domain-specific corpora but also on the general Bnc corpus. Note that the most frequent sense in the test data may be considered as an upperbound for domain adaptation, as it requires tagging examples

	Similar			Different		
Systems	Bnc	Sp.	Fin.	Bnc	Sp.	Fin.
SemCor MFS	54.7	**65.5**	**79.0**	9.7	3.8	8.4
k-NN	**57.1**	64.6	69.9	24.6	18.5	25.4
Context PPR	50.0	34.9	64.2	**36.0**	35.9	35.0
Related PPR	38.1	53.1	73.7	24.8	**50.9**	**49.5**

Table VII.2: Results for those words with similar (and different) sense distributions. Best results in bold.

from each target domain. The supervised systems scarcely improve over the SemCor MFS, which is consistent with state-of-the-art results over all-words datasets (Snyder and Palmer, 2004; Pradhan *et al.*, 2007). They also lie well below the Test MFS, with a dramatic gap in the two domain corpora. The low results on the Bnc, even being a general corpora, show that the deployment of supervised systems is problematic, not only because of domain shifts, but also when being applied to different corpora, even being both general domain, as already attested in the literature (Escudero *et al.*, 2000).

Static PageRank: Applying PageRank over the entire WordNet graph yields low results, very similar to those of SemCor MFS, and below those of all Personalized PageRank variants that we tried. In fact, Static PageRank seems to be closely related to the SemCor MFS, as we hypothesized in Section VII.3.1.

Personalized PageRank over context words: Surprisingly, applying our Personalized PageRank method for each occurrence yields results which are above the supervised systems in all three corpora, with larger improvements for the domain-specific ones. The results of the strategies for selecting one single sense as output (*maxsense* or *all-in-one*) are mixed, with slight improvements in Sports and Finance and degradation in the Bnc.

Personalized PageRank over related words: Personalized PageRank over related words obtains the best results overall for Sports and Finance, and it is thus a preferred strategy to disambiguate domain-specific words. Interestingly, in the case of the balanced Bnc corpus, the best results are for Personalized PageRank over the occurrence context. It seems that using related words is optimal for domain-specific WSD, but not for general purpose WSD, where a more personalized case-by-case treatment is required for each occurrence. Finally, the combination of the occurrence context and related words does not seem to be productive, as attested by its decrease in performance.

Comparison to Koeling *et al.* (2005): Our Personalized PageRank algorithm over related words performs significantly better than Koeling *et al.* (2005) in both SPORTS and FINANCE. Our WSD method is closely related to the WordNet-based similarity method defined in (Hughes and Ramage, 2007). In this sense, our WSD algorithm is an alternative to the one used in (Koeling *et al.*, 2005). Instead of computing pairwise similarity and selecting the sense which yields the maximum additive similarity score with respect to each related words from the thesaurus, our WSD method implicitly yields the sense with the maximum similarity score with respect to the full set of related words in one go. In fact, we initialize PPR with all the related words in the thesaurus, yielding the sense of the target word with the highest rank, i.e. the sense which is most closely related to those words. Our better results are consistent with the word similarity results reported in (Hughes and Ramage, 2007), which surpass other WordNet-based similarity methods, including those used by Koeling *et al.* (2005).

Results for coarse-grained senses: WordNet senses have been criticized for their fine-grainedness. Public evaluation exercises have also used coarse-grained senses as defined by the semantic files of the senses. In our case, the best results for BNC, SPORTS and FINANCE measured according to coarse senses would be of 61.2%, 72.0% and 56.9%, respectively. This is remarkable given the high polysemy of the target words. The overall results for SPORTS and FINANCE are higher than for BNC, holding promise for building high performance domain-specific WSD in specific domains.

Effect of sense distributions: The performance of the supervised systems could be affected by the very different distribution of word senses in the training (SemCor) and test dataset (all three corpora), as attested by the very low performance of the SemCor MFS and the dramatic difference with respect to the Test MFS. We decided to make two groups of words for each corpus, according to the similarity of sense distributions measured as the difference between SemCor MFS and Test MFS. In the *Similar distribution* group we included those words with differences below 10 percentage points, and in the *Different distribution* group those with larger differences. Note that for each domain we acquire different set words: for SPORTS we had 12 *Similar* words and 29 *Different* words, for FINANCE we had 16 and 25, and for the BNC 19 and 22, respectively. Table VII.2 shows that for *Similar* sense distributions the best results are actually those of the SemCor MFS and the

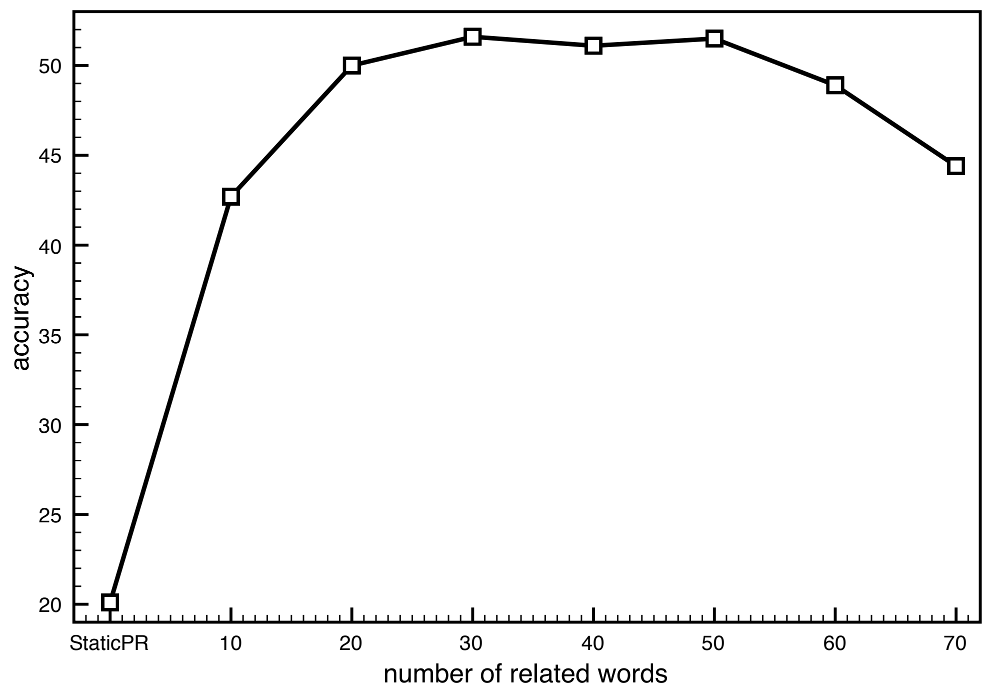

Figure VII.1: Learning curves for PPR (SPORTS) using different numbers of related words from the thesaurus. Leftmost point corresponds to zero words, which equals to static PR.

supervised WSD system, while the Personalized PageRank algorithms yield the best results for *Different* sense distributions.

Exploring the number of related words: Following (Koeling *et al.*, 2005) we used the 50 most similar words when doing WSD. Figure VII.1 shows the number of words is an important factor, with best results for 30-50 words. These results agree with the intuition that a minimum amount of words is necessary for providing context, but introducing further terms down the list of related words involves noisier and less related words.

VII.5 Conclusions and future work

This chapter shows that Knowledge-Based WSD systems are a powerful alternative to supervised WSD systems when tagging domain-specific corpora.

The results range from 51.5% to 59.3% (61.2% to 72% coarse-grained) for 41 domain-related and highly polysemous words. Interestingly, the results are higher for domain-specific corpus than for the general corpus, raising interesting prospects for improving current WSD systems when applied to specific domains.

The results also show that our knowledge-based WSD algorithm improves significantly over previous results on the same dataset. The system and the data are publicly available in `http://ixa2.si.ehu.es/ukb/`. Disambiguating related words from an automatically built thesauri (instead of the actual occurrence contexts) yields better results on the domain datasets, but not on the general one.

Our analysis showed that the differences in sense distribution hurt supervised systems, and a combination of supervised and knowledge-based systems which takes this into account seems promising. In particular, our approach is complementary to supervised domain-adaptation techniques (Chapter V and Chapter VI). Given the existing difference with respect to the Test MFS, there is ample room for further improvements.

The dataset we used is a lexical-sample, and our results might depend on the actual words in the sample. For the future, we would like to confirm our findings on an all-words domain-specific corpus.

CHAPTER VIII

Conclusions and future work

In the introductory chapter we stated that Word Sense Disambiguation is one of the most important problems in NLP semantics. But in order to be useful, a WSD classifier has to be robust and noise-tolerant to maintain its performance across texts, and has also to be easy to adapt to new domains. WSD classifiers based on ML usually suffer from **data sparseness**, and suffer in **domain shifts**.

This dissertation has focused on data sparseness and specially domain shifts. Although our emphasis has been on developing a technique for domain adaptation of supervised systems, we have also experimented with a knowledge-based algorithm. In order to better understand the issues that arise when facing domain shifts, we can mention the following closely related points for word senses: 1) **word sense distributions might change drastically**, for example, in the sports domain the meaning of *coach* is more likely to be *someone in charge of training an athlete or a team* rather than a specific type of vehicle; 2) **word sense variability decreases in domain-specific corpora**, that is words tend to occur in less senses, where the probability of the predominant sense is usually higher (high *skew*) and the rest is lower, and can disappear; 3) **new word senses might arise in domain-specific corpora**, that is, specialized uses of domain words might create new senses.

Similar issues arise with the features used for learning. The words in context change across domains, as well as syntax and structures of phrases. Thus, extracted features would be different, and algorithms would generalize differently. In other words, we say that the **feature distribution** also changes across domains.

In order to alleviate all the above problems, this dissertation proposed to use **Singular Value Decomposition** (SVD). SVD has been shown to mitigate the data sparseness problem by finding condensed representations of the task and reducing significantly the dimensionality of the feature space. In domain adaptation SVD helps to reduce the gap between feature distributions of both source and target domains throwing the new features into same distribution. The feature space represented by SVD helps the ML algorithm to induce good models and, thus, select correct word sense, even if the sense distributions keeps being different in both domains. In order to decrease the gap between source and target domain, we also apply **semi-supervised learning**, introducing unlabeled data in the learning process.

The combination of the different features has been shown to be key factor to obtain robustness and domain adaptation in WSD. This dissertation has presented a novel way to combine k-NN classifiers, showing its robustness across several datasets. In addition, we show that the use of the kernels is an elegant way and effective way to combine the feature spaces.

This dissertation has followed a clear experimental strategy. We first started in a general domain, without paying attention to any cross domain problem. In these experiments, reported in Chapter IV we focused on data sparseness. The next two sets of experiments were devoted to domain adaptation issues. In Chapter V semi-supervised domain adaptation was addressed, using solely unlabeled data in order to adapt the system to the new domain. In Chapter VI supervised domain adaptation experiments were carried out, where both source and target examples are used to train the WSD classifier. In addition, unlabeled data was used to improve the adaptation results.

Additional experiments, related to domain-specific WSD, were carried out in Chapter VII. There we showed how knowledge-based systems can perform as well as (and even better than) supervised systems, due to the fact that supervised ML systems depend significantly on the corpus bias. These are very encouraging results, and the combination of both paradigms can be key for future developments.

From the experiments on each of the aspects of WSD treated in this dissertation, we were able to extract some conclusions. We will try first to summarize what we consider the main contributions of this dissertation, and then we will describe the conclusions derived from each of the studied issues.

VIII.1 Contribution of our work

We will now describe the main contributions of this work, which were already advanced in Chapter I, including their relation to chapters in the dissertation:

- **SVD decomposition is useful to deal with data sparseness and domain adaptation** (Chapters 4, 5 and 6): We explored the contribution of features obtained with SVD decomposition to WSD performance. We presented several experiments across different datasets. We studied the performance of a number of ML algorithms trained on these types of features and analyzed the effect of the number of dimensions. We also developed different ways to obtain the SVD representation, each catching different evidences from text. The SVD representation is complementary to the *classic* feature set, and we showed that combining them is a robust way to improve the results. We used two experimental scenarios: general domain WSD was tested in Senseval and SemEval datasets, and domain-specific WSD. Our results obtain the state-of-the-art over Senseval-like datasets. In domain adaptation, we showed that SVD features are good enough to obtain robustness (on a general WSD system) and adaptation (on a system trained with examples from general domain and domain-specific instances).

- **Unlabeled data helps find better correlations among features** (Chapters 4, 5 and 6): We studied the usefulness of unlabeled data to obtain better correlation among the features from labeled instances. We use unlabeled data to help find higher-order correlations applying SVD to the augmented matrix. This matrix augmentation is also known as *background learning*. In order to asses the effect of the unlabeled data we evaluated WSD systems trained on different amounts of unlabeled data. We reported several experiments showing that unlabeled data help up to certain amounts. On the domain adaptation scenario, we played with unlabeled data from different sources, finding that unlabeled data must be topically related to the training set in order to obtain effective SVD features. We showed that unlabeled data helps obtain more reliable features and an it is an important factor in the domain adaptation scenario.

- **Combination of feature spaces is useful** (Chapters 4 and 6): the redundancy and heterogeneity of features can affect negatively. We split the original feature set in coherent sets, and explored how to combine them. In k-NN combination each k-NN system is trained on a different feature space and casts votes for its first k neighbors. In kernel-based combination, each kernel's implicit function is a different kind of SVD mapping. The former method obtain the state-of-the-art results in Senseval dataset. The latter showed that it is an effective way to take profit of the general source domain in the supervised domain adaptation scenario. We show that a combination of rich feature spaces is a useful and robust manner to obtain good results for WSD.

- **Robustness in face of semi-supervised domain adaptation** (Chapter 5). Using SVD and unlabeled data we obtained a robust system that performs well across different target domains without labeled data from the target domains, reducing the domain shift problem for general WSD system. We found that in order to obtain a robust system the unlabeled data should be from general domain and be related to the training set.

- **Supervised domain adaptation** (Chapter 6). We have showed for the first time that source general domain examples are an useful addition to target domain examples in WSD, and provide for the best results domain adaptation. Up to now, the general domain examples were not shown to be useful. We concluded that the correlations found by SVD and the generalization provided by the combination of SVD-based kernels are effective.

In order to make a more complete picture of domain adaptation, we also explored the performance of knowledge-based models:

- **Knowledge-based WSD system may outperform a general WSD system** (Chapter 7): We explored the application of knowledge-based WSD systems to specific domains, based on a combination of state-of-the-art graph-based WSD system (Agirre and Soroa, 2009) that uses the information in WordNet with a distributional thesaurus built from the target domains. This system outperformed supervised systems trained on SemCor, showing that knowledge-based WSD systems are a powerful alternative to supervised WSD systems.

VIII.2 Conclusions

In order to achieve the results described in the previous section, we follow a path through different WSD issues, which serves 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.

Combination of features spaces and unlabeled data with WSD (4th chapter)

In this chapter we deal with data sparseness. We tried three ideas to improve the WSD performance:

- The use of SVD to deal with data sparsity and redundancy.
- The use of unlabeled data in order to improve the learning process.
- The combination of various sets of features in order to improve the classifiers.

We applied SVD to two different kinds of matrices. Our experiments show that the OMT technique to apply SVD compares favorably to SMA, which has been previously used in (Gliozzo *et al.*, 2005). Although constructing one matrix per target word (OMT) yields the best results, it is a relatively expensive process, so we did not apply it to the all-words setting. In the all-words experiment we only tested the single matrix for all (SMA) approach, with good results. The results show that our combined k-NN systems are state-of-the-art, specially in lexical sample settings, and that the induced features provide significant improvements.

We performed several sets of experiments in order to explore different feature spaces and combination. Next we will describe those conclusions more in detail:

OMT **, unlabeled data and feature split** (*first set of experiments*). In this experiments we based on SVD features induced from OMT matrices, unlabeled data to provide *background knowledge*, and splits of original feature spaces to tackle the sparseness, redundancy and heterogeneity in data. Each of the proposals improves the results for a k-NN classifier, and properly combined they provide we obtained one of the best results for the Senseval-3 lexical-sample dataset.

We argued that these improvements help to model better the feature space, which, coupled with a ML algorithm well suited for combination such as k-NN, explains the good results. This opens new feature modeling possibilities.

SMA **features set and combinations** (*Second set of experiments*). This set of experiments explored the split of feature sets in order to obtain better WSD systems through combinations of classifiers learned over each of the split feature sets. Our results show that k-NN is able to profit from the combination of split features (contrary to VSM and SVM), and that simple voting is not enough for that. Instead we propose combining all k-NN subsystems where each of the k neighbors casts one vote.

We showed that SMA features behaved well through different datasets. The experiments demonstrated that SMA and the combination with the original features finds different patterns in text, improving consistently the system performances.

We have performed a thorough evaluation on two datasets (Senseval-3 lexical-sample and all-words), having promising results.

Participation in Semeval-2007 (*third set of experiments*). In these experiments we confirmed our findings in the previous experiments. First of all, we built a robust system based on a number of k-NN classifiers , each trained on a different feature spaces. The classifier was tested in lexical-sample and all-words tasks. Regarding lexical-sample, we saw that OMT features outperform the rest of the features, and that SMA features are useful in combinations.

Semi-supervised domain adaptation (5th chapter)

In this chapter we explore robustness and domain adaptation issues for WSD using SVD and unlabeled data. We focus on the semi-supervised scenario, where we train on the source corpus (BNC), test on two target corpora (SPORTS and FINANCE sections of Reuters), and improve the results using unlabeled data.

Our method yields up to 16.3% error reduction compared to SVM and k-NN on the labeled data alone, showing the first positive results on domain adaptation for WSD. In fact, we show that our results are due to the use of a large, general, unlabeled corpus, and rather than domain-adaptation proper we show robustness in face of a domain shift. This kind of robustness is even more cost-effective than semi-supervised domain adaptation, as it does not require large unlabeled corpora or repeating the computations for each new target domain.

These experiments show that the OMT technique to apply SVD that compares favorably to SMA. We also show that the unlabeled data needs to be related to the training data, and that the benefits of our method are larger when faced with a domain shift (compared to test data coming from the same domain as the training data).

Supervised domain adaptation (6th chapter)

In this chapter we explored supervised domain adaptation for WSD with positive results, showing that hand-labels from the general domain (source text) are useful when training a WSD system that is be applied to specific domains (target texts). We performed several experiments in three scenarios. In the first scenario (source scenario), the classifiers were trained on general source domain data (the BNC) and tested on the target domains, composed by the SPORTS and FINANCE sections of Reuters. In the second scenario (target scenario) we set the main baseline for our domain adaptation experiment, training and testing our classifiers on the target domain data. In the last scenario (domain adaptation scenario), we combined both source and target data for training, and test on the target data.

Our results show that our best domain adaptation strategy (using kernel combination of SVD features and unlabeled data related to the training data) yields statistically significant improvements: up to 22% error reduction compared to SVM on the target domain data alone. We also show that our domain adaptation method only needs 40% of the target data (in addition to the source data) in order to get the same results as SVM on the target alone.

We obtained coherent results in two target scenarios, and consistent improvement at all levels of the learning curves, showing the robustness of our findings. We think that our dataset, which comprises examples for 41 nouns that are closely related to the target domains, is specially demanding, as one would expect the performance of a generic WSD system to drop when moving to the domain corpus, specially for domain-related words, while we could expect the performance to be similar for generic or unrelated words.

Knowledge-based domain-specific WSD (7th chapter)

This chapter showed that Knowledge-Based WSD systems are a powerful alternative to supervised WSD systems when tagging domain-specific corpora. The results range from 51.5% to 59.3% (61.2% to 72% coarse-grained) for 41 domain-related and highly polysemous words. Interestingly, the results are higher for domain-specific corpus than for the general corpus, raising interesting prospects for improving current WSD systems when applied to specific domains.

The results also showed that our knowledge-based WSD algorithm improves significantly over previous results on the same dataset. Disambiguating related words from an automatically built thesauri (instead of the actual occurrence contexts) yields better results on the domain datasets, but not on the general one.

Our analysis showed that the differences in sense distribution hurt supervised systems, and a combination of supervised and knowledge-based systems which takes this into account seems promising. In particular, our approach is complementary to supervised domain-adaptation techniques (Chapter V and Chapter VI). Given the existing difference with respect to the Test MFS, there is ample room for further improvements.

VIII.3 Future Work

There are some open research lines in this work that can be explored further. We will describe the main experiments and path to be explored we would like perform in the future.

- **It would be interesting to test our findings in other datasets**, such as DSO or OntoNotes, and see if the positive results are confirmed. We would also like to study word-by-word behavior, in order to asses whether target examples are really necessary for words which are less related to the domain. Following the experiments in Chapter VII we would like to test whether our results depend on the actual words in the sample.

- **Find valid representations for cross-domain problems**. In terms of features, we think that some kind of features might not depend so much on the domain. There is previous work that tried to define this kind of specific features, where they can be used as a bridge to transfer the useful knowledge in one domain to another (Blitzer *et al.*, 2006). Unfortunately, there is no published work that takes WSD as a problem.

- **SemEval 2010: All-words Word Sense Disambiguation on a specific domain**. In order to perform a proper analysis of the domain adaptation issue, we are organizing an all-words task to perform experiments in a domain-specific corpus. This way, we could confirm our findings in this dissertation. The task is described more in detail in `http://xmlgroup.iit.cnr.it/SemEval2010/`.

- **Adaptation of Knowledge-based approaches**. This is closely related to Ontology learning, where we would like to adapt an ontology (or a knowledge base) by pruning and adding concepts belonging to the specific-domain. This way, better WSD could be achieved.

- **Combination of Knowledge-based and supervised approaches**. In general terms, WSD has a long history, and it seems that supervised approaches have reached to a plateau state. Chapter VII has shown the weakness of supervised algorithms, as the difference in the sense distribution hurts supervised systems. A combination of supervised and knowledge-based systems which take into account the sense distribution seems promising.

- **Indirect evaluation of WSD systems**. Important advances have been introduced in WSD task since early 80's in terms of recall and precision, but still there is loads of work to integrate as useful module in other task such as IR or MT. We would like to investigate how this semantic knowledge could be integrated further in NLP and HLT applications.

Bibliography

Agirre E., Baldwin T. and Martínez D. *Improving parsing and pp attachment performance with sense information*. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL HLT 2008)*, pp. pp. 317–325. Columbus, USA, 2008.

Agirre E. and Edmonds P., eds. *Word Sense Disambiguation. Algorithms and Application*, volume 33 of *Text, Speech and Language Technology*. Springer, P.O. Box 17, 3300 AA Dordrecht, The Netherlands, 2006.

Agirre E. and Martínez D. *Knowledge sources for word sense disambiguation*. In V. Matousek, P. Mautner, R. Moucek and K. Tauser, eds., *Proceedings of the Fourth International Conference TSD*, Published in the Springer Verlag Lecture Notes in Computer Science series. Copyright Springer-Verlag, 2001a.

Agirre E. and Martínez D. *Learning class-to-class selectional preferences*. In *Proceedings of the 5th Conference on Natural Language Learning*, pp. 15–22, 2001b.

Agirre E. and Martínez D. *Smoothing and Word Sense Disambiguation*. In *Proceedings of EsTAL - España for Natural Language Processing*. Alicante, Spain, 2004a.

Agirre E. and Martínez D. *The effect of bias on an automatically-built word sense corpus*. Proceedings of the 4rd International Conference on Languages Resources and Evaluations (LREC), 2004b.

Agirre E. and Rigau G. *Word sense disambiguation using conceptual density*. In *Proceedings of 16th International Conference on Computational Linguistics (COLING-96)*, pp. 12–16, 1996.

Agirre E. and Soroa A. *Using the multilingual central repository for graph-based word sense disambiguation*. In *Proceedings of LREC '08*. Marrakesh, Morocco, 2008.

Agirre E. and Soroa A. *Personalizing pagerank for word sense disambiguation*. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL-09)*. Athens, Greece, 2009.

Alshawi H. and Carter D. *Training and scaling preference functions for disambiguation*. Computational Linguistics, volume 20(4), pp. 635–648, 1994.

Ando R. and Zhang T. *A framework for learning predictive structure from multiple tasks and unlabeled data*. Journal of Machine Learning Research, volume 6, pp. 1817—1853, 2005.

Ando R.K. *Applying alternating structure optimization to word sense disambiguation*. In *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL)*, pp. 77–84. New York City, 2006.

Atkins S. *Tools for Computer-Aided Corpus Lexicography: The Hector Project*, 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.

Bay S.D. *Nearest neighbor classification from multiple feature subsets.*. Intell. Data Anal., volume 3(3), pp. 191–209, 1999.

Ben-David S., Blitzer J., Crammer K. and Pereira F. *Analysis of representations for domain adaptation*. In B. Schölkopf, J. Platt and T. Hoffman, eds., *Advances in Neural Information Processing Systems 19*, pp. 137–144. MIT Press, Cambridge, MA, 2007.

Blitzer J., McDonald R. and Pereira F. *Domain adaptation with structural correspondence learning*. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pp. 120–128. Association for Computational Linguistics, Sydney, Australia, 2006.

Brin S. and Page L. *The anatomy of a large-scale hypertextual web search engine*. Computer Networks and ISDN Systems, volume 30(1-7), 1998.

Brockmann C. and Lapata M. *Evaluating and combinin approaches to selectional preference acquisition*. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL-03)*, pp. 27–34, 2003.

Brown P.F., Pietra S.D., Pietra V.J.D. and Mercer R.L. *Word-sense disambiguation using statistical methods*. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 264–270. Berkeley, California, 1991.

Cai J.F., Lee W.S. and Teh Y.W. *Nus-ml:improving word sense disambiguation using topic features*. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pp. 249–252. Association for Computational Linguistics, Prague, Czech Republic, 2007.

Carpuat M. and Wu D. *Evaluating the word sense disambiguation performance of statistical machine translation*. In *Second International Joint Conference on Natural Language Processing (IJCNLP-2005)*, 2005.

Chan Y.S. and Ng H.T. *Word sense disambiguation with distribution estimation.*. In *Proceeings of the 19th International Joint Conference on Artificial Intelligence (ings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1010—1015. Edingurgh, Scotland, 2005.

Chan Y.S. and Ng H.T. *Estimating class priors in domain adaptation for word sense disambigua- tion*. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING-ACL)*, pp. 89–96. Sydney, Australia, 2006.

Chan Y.S. and Ng H.T. *Domain adaptation with active learning for word sense disambiguation*. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 49–56. Association for Computational Linguistics, Prague, Czech Republic, 2007.

Chan Y.S., Ng H.T. and Chiang D. *Word sense disambiguation improves statistical machine translation*. In *Proceedings of the 45th Annual Meeting* *of the Association of Computational Linguistics*, pp. 33–40. Association for Computational Linguistics, Prague, Czech Republic, 2007a.

Chan Y.S., Ng H.T. and Zhong Z. *Nus-pt: Exploiting parallel texts for word sense disambiguation in the english all-words tasks*. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pp. 253–256. Association for Computational Linguistics, Prague, Czech Republic, 2007b.

Chapellea O., Schölkopf B. and Zien A. *Semi-supervised learning*. MIT press, 2006.

Chapman R.L. *Roget's International Thesaurus, Fourth Edition*. Harper and Row, New York, USA, 1977.

Chelba C. and Acero A. *Adaptation of maximum entropy classifier: Little data can help a lot*. In *Proceedings of of th Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Barcelona, Spain, 2004.

Clough P. and Stevenson M. *Cross-language information retrieval using eurowordnet and word sense disambiguation*. In *Advances in Information Retrieval, 26th European Conference on IR Research (ECIR)*, pp. 327–337. Sunderland, UK, 2004.

Cristianini N. and Shawe-Taylor J. *Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, 2000.

Cruse A.D. *Lexical Semantics*. Cambrige University Press, Cambridge, UK, 1986.

Daelemans W. and van den Bosch A. *Memory-based language processing*. Studies in Natural Language Processing. Cambrige University Press, 2005.

Daelemans W., van den Bosch A. and Zavrel J. *Forgeting exceptions is harmful in language learning*. Machine Learning, volume 34, pp. 11–41, 1999.

Daelemans W., Zavrel J., van der Sloot K. and van den Bosch A. *Timbl: Tilburg memory based learner, version 6.1, reference guide*. Technical report, ILK Research Group Technical Report, 2007.

Dai W., Xue G.R., Yang Q. and Yu Y. *Boosting for transfer learning*. In *Proceedings of the 24th Annual International Conference on Machine Learning*, pp. 193–200. Corvallis, Oregon, USA, 2007a.

Dai W., Xue G.R., Yang Q. and Yu Y. *Transferring naive bayes classifiers for text classification.* In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pp. 540–545. Vancouver, Canada, 2007b.

Daude J., Padro L. and Rigau G. *Mapping wordnets using structural information.* In *Proceedings of 38th Anual Meeting of the Association for Computational Linguistics*. Hong Kong, PRC, 2000.

Daumé III H. *Frustratingly easy domain adaptation.* In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 256–263. Association for Computational Linguistics, Prague, Czech Republic, 2007.

Daumé III H. and Marcu D. *Domain adaptation for statistical classifiers.* Journal of Artificial Intelligence Research, volume 26, pp. 101–126, 2006.

Decadt B., Hoste V., Daelemans W. and van den Bosch A. *Gambl, genetic algorithm optimization of memory-based wsd..* In R. Mihalcea and P. Edmonds, eds., *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text (Senseval-3)*, pp. 108–112. ACL, Barcelona, Spain, 2004.

Deerwester S., Dumais S., Furnas G., Landauer T. and Harshman R. *Indexing by Latent Semantic Analysis.* Journal of the American Society for Information Science, volume 41(6), pp. 391–407, 1990.

Dietterich T.G. *Machine-learning research: Four current directions.* The AI Magazine, volume 18(4), pp. 97–136, 1997.

Eckart C. and Young G. *The approximation of one matrix of another of lower rank.* Psychometrika, volume 1, pp. 211–218, 1936.

Edmonds P. and Cotton S. *SENSEVAL-2: Overview.* In *Proceedings of the Second International Workshop on evaluating Word Sense Disambiguation Systems.*. Toulouse, France, 2001.

Escudero G., Márquez L. and Rigau G. *An Empirical Study of the Domain Dependence of Supervised Word Sense Didanbiguation Systems*. Proceedings of the joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP/VLC, 2000.

Fellbaum C. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

Florian R., Cucerzan S., Schafer C. and Yarowsky D. *Combining classifiers for word sense disambiguation*. Natural Language Engineering, volume 4(8), pp. 327–341, 2002.

Francesca F. and Zanzotto F.M. *Svd feature selection for probabilistic taxonomy learning*. In *Proceedings of EACL 2009 Workshop on GEMS: GEometrical Models of Natural Language Semantics*, pp. 66–73. Athens, Greece, 2009.

Gliozzo A.M., Giuliano C. and Strapparava C. *Domain Kernels for Word Sense Disambiguation*. 43nd Annual Meeting of the Association for Computational Linguistics. (ACL-05), 2005.

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.

Harris Z. *Mathematical structures of language*. Interscience Publisher, New York, 1968.

Haveliwala T.H. *Topic-sensitive pagerank*. In *WWW '02: Proceedings of the 11th international conference on World Wide Web*, pp. 517–526. ACM, New York, NY, USA, 2002.

Hirst G. and St-Onge D. *Lexical chains as represetations of context in the detection and correction of malapropisms*. In C. Fellbaum, ed., *WordNet: An Electronic Lexical Database*, Lexical chains as represetations of context in the detection and correction of malapropisms, pp. 305–332. MIT Press, Massachusetts, USA, 1998.

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 Second International Workshop on Evaluating Word Sense Disambiguation Systems (Senseval-2)*, pp. 83–86. Toulouse, France, 2001.

Hovy E., Marcus M., Palmer M., Ramshaw L. and Weischedel R. *Ontonotes: The 90% solution*. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pp. 57–60. Association for Computational Linguistics, New York City, USA, 2006.

Hughes T. and Ramage D. *Lexical semantic relatedness with random graph walks*. In *Proceedings of EMNLP-CoNLL-2007*, pp. 581–589, 2007.

Ide N. and Véronis J. *Word sense disambiguation: The state of the art*. Computational Linguistics, volume 24(1), pp. 1–40, 1998.

Jiang J. and Zhai C. *Instance weighting for domain adaptation in nlp*. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*, pp. 254—271. Prague, Czech Republic, 2007a.

Jiang J. and Zhai C. *A two-stage approach to domain adaptation for statistical classifiers*. In *Proceedings of the ACM 16th Conference on Information and Knowledge Management*, pp. 401—410, 2007b.

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.

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.

Killgariff A. *I don't believe in word senses*. Computers and the Humanities, volume 31, pp. 91–113, 1997.

Killgariff A. and Tugwell D. *Sketching words*. In M.H. Corréard, ed., *Lexicography and Natural Language Processing: A Festschrift in Honour of B. T. S. Atkins*, chapter Sketching words, pp. 125–137. EURALEX, 2004.

Kim H., Howland P., and Park H. *Dimension reduction in text classification with support vector machines*. Journal of Machine Learning Research, volume 6, pp. 37–53, 2005.

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

Kohomban U.S. and Lee W.S. *Learning Semantic Classes for Word Sense Disambiguation*. In *43nd Annual Meeting of the Association for Computational Linguistics. (ACL-05)*. University of Michigan, Ann Arbor, 2005.

Landauer T.K. and Dumais S.T. *A solution to plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge*. Psychological Review, volume 104, pp. 211–240, 1997.

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.

Leacock C., Towell G. and Voorhees E. *Toward building contextual representations of word senses using statistical models*. In *Proceedings of ACL SIGLEX Workshop on Acquisition of Lexical Knowledge from Text*, pp. 10–20, 1993.

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

Lesk M. *Automatic sense disambiguation using machine readable dictionaries: How to tell a pine come from a ice cream cone*. In *Proceedings of SIGDOC-86: 5th International Conference on Systems Documentation*, pp. 24–26. Toronto, Canada, 1986.

Lin D. *Automatic retrieval and clustering of similar words*. In *Proceedings of the 36th Annual Meeting of the Association of Computational Linguistics (ACL98) - joint with Computational Linguistics (Coling98)*. Montreal, Canada, 1998.

Lyons J. *Linguistic Semantics: An Introduction*. Cambrige University Press, 1995.

Magnini B. and Cavagliá G. *Integrating subject field codes into WordNet*. In *Proceedings of the Second International LREC Conference*. Athens, Greece, 2000.

Mann H.B. and Whitney D.R. *On a test of whether one of two random variables is stochastically larger than the other*. Annals of Mathematical Statistics, volume 18, pp. 50–60, 1947.

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

Martínez D. *Supervised Word Sense Disambiguation: Facing Current Challenges*. Ph.D. thesis, informatika Fakultatea, UPV-EHU, 2004.

Martínez D. and Agirre E. *One Sense per Collocation and Genre/Topic Variations*. Conference on Empirical Method in Natural Language, 2000.

McCarthy D., Koeling R., Weeds J. and Carroll J. *Finding predominant word senses in untagged text*. In *ACL '04: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, p. 279. Association for Computational Linguistics, Morristown, NJ, USA, 2004.

McCarthy D., Koeling R., Weeds J. and Carroll J. *Unsupervised acquisition of predominant word senses*. Computational Linguistics, volume 33(4), 2007.

McRoy S.W. *Using multiple knowledge source for word sense disambiguation*. Computational Linguistics, volume 18, pp. 1–30, 1992.

Mihalcea R. *Bootstrapping large sense tagged corpora*. In *Proceeding of the Third International Conference on Language Resources and Evaluation (LREC-02)*. Las Palmas, Canary Islands, Spain, 2002.

Mihalcea R. *Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling*. In *Proceedings of HLT05*. Morristown, NJ, USA, 2005.

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. *A mehthod for word sense disambiguation*. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL 1999),*, pp. 152–158, 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. and Charles W. *Contextual correlates of semantic similarity*. Language and Cognitive Processes, volume 6(1), pp. 1–26, 1991.

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., Leacock C., Tengi R. and R.Bunker. *A Semantic Concordance*. In *Proceedings of the ARPA Human Language Technology Workshop. Distributed as* Human Language Technology *by San Mateo, CA: Morgan Kaufmann Publishers.*, pp. 303–308. Princeton, NJ, 1993.

Navigli R. and Lapata M. *Graph connectivity measures for unsupervised word sense disambiguation*. In *IJCAI*, 2007.

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.

Ng H.T. and Lee H.B. *Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach*. In *Proceedings of the 34th Annual Meeting of the Association for Computationla Linguistics (ACL)*, pp. pp. 40–47, 1996.

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.

Niu Z.Y., Ji D.H. and Tan C.L. *Word sense disambiguation using label propagation based semi-supervised learning*. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pp. 395–402. Association for Computational Linguistics, Ann Arbor, Michigan, 2005.

Niu Z.Y., Ji D.H. and Tan C.L. *I2r: Three systems for word sense discrimination, chinese word sense disambiguation, and english word sense disambiguation*. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pp. 177–182. Association for Computational Linguistics, Prague, Czech Republic, 2007.

Noreen E.W. *Computer-Intensive Methods for Testing Hypotheses*. John Wiley & Sons, 1989.

Otegi A., Agirre E. and Rigau G. *Ixa at clef 2008 robust-wsd task: using word sense disambiguation for (cross lingual) information retrieval*. In *Working Notes of the Cross-Lingual Evaluation Forum*, 2008.

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.

Pang B. and Lee L. *A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts*. In *Proceedings of 42nd Anual Meeting of the Association for Computational Linguistics (ACL-04)*, pp. 271–278. Barcelona, Spain, 2004.

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.

Pereira F. and Gordon G. *The support vector decomposition machine*. In *Proceedings of the 23rd international conference on Machine learning*. Pittsburgh, PA, USA, 2006.

Pradhan S., Loper E., Dligach D. and Palmer M. *Semeval-2007 task-17: English lexical sample, srl and all words*. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pp. 87–92. Prague, Czech Republic, 2007.

Resnik P. *Using information content to evaluate semantic similarity*. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 448–553, 1995.

Rose T.G., Stevenson M. and Whitehead M. *The reuters corpus volumen 1 – from yesterday's news to tomorrow's language resources*. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC-2002)*, pp. 827–832. Las Palmas, Canary Islands, 2002.

Satpal S. and Sarawagi S. *Domain adaptation of conditional probability models via feature subsetting*. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 224–235. Warsaw, Poland, 2007.

Schütze H. *Automatic word sense discrimination*. Computational Linguistics, volume 24(1), 1998.

Shimodaira H. *Improving predictive inference under covariate shift by weighting the log-likelihood function*. Journal of Statistical Planning and Inference, volume 2(90), pp. 227–244, 2000.

Sinha R. and Mihalcea R. *Unsupervised graph-based word sense disambiguation using measures of word semantic similarity*. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC 2007)*. Irvine, CA, USA, 2007.

Snow R., Jurafsky D. and Ng A.Y. *Semantic taxonomy induction from heterogenous evidence*. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pp. 801–808. Association for Computational Linguistics, Sydney, Australia, 2006.

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.

Stephan B. and Hotho A. *Text classification by boosting weak learners based on terms and concepts*. In *Proceedings of the Fourth IEEE International Conference on Data Mining*, pp. 331–334, 2004.

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

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

Storkey A.J. and Sugiyama M. *Mixture regression for covariate shift*. In B. Schölkopf, J. Platt and T. Hoffman, eds., *Advances in Neural Information Processing Systems 19*, pp. 1337–1344. MIT Press, Cambridge, MA, 2007.

Strapparava C., Gliozzo A.M. and Giuliano C. *Pattern abstraction and term similarity for 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.

Su W., Wu D. and Carpuat M. *Semi-Supervised Training of a Kernel PCA-Based Model for Word Sense Disambiguation*. 20th International Conference on Computational Linguistics (COLING-2004), 2004.

Thomasian A., Li Y. and Zhang L. *Exact k-NN queries on clustered SVD datasets*. Information Processing Letters (ILP), volume 94, pp. 247–252, 2005.

Tratz S., Sanfilippo A., Gregory M., Chappell A., Posse C. and Whitney P. *Pnnl: A supervised maximum entropy approach to word sense disambiguation*. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pp. 264–267. Association for Computational Linguistics, Prague, Czech Republic, 2007.

Tuggy D. *Ambiguity, polysemy, and vagueness*. Cognitive Linguistics, volume 4(3), pp. 273–290, 1993.

Vapnik V. *Statistical Learning Theory*. John Wiley & Sons, 1998.

Vapnik V.N. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

Vossen P. *EuroWordNet: A Multilingual Database with Lexical Semantic Networks*. Kluwer Academic Publishers, Dordrecht, 1998.

Vossen P., Rigau G., Alegria I., NewAuthor4, Farwell D. and Fuentes M. *Meaningful results for information retrieval in the meaning project*. In *Proceedings of the 3rd Global Wordnet Conference*. Jeju Island, Korea, 2006.

Weber M., Mork J.G. and Aronson A.R. *Developing a test collection for biomedical word sense disambiguation*. In *Proceedings of AMAI Symposium)*, pp. pp. 746–750. Washington, DC, 2001.

Wilks Y. *Making preferences more active*. Artificial Intelligence, volume 11(3), pp. 197–223, 1978.

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

Wu D., Su W. and Carpuat M. *A Kernel PCA Method for Superior Word Sense Disanbiguation*. In *42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*. Barcelona, Spain, 2004.

Xing D., Dai W., Xue G.R. and Yu Y. *Bridged refinement for transfer learning*. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 324—335. Warsaw, Poland, 2007.

Yarowsky D. *Statistical models of roget's categories trained on large corpora*. In *Proceedings of 14th International Conference on Computational Linguistics (COLING-92)*, 1992.

Yarowsky D., Cucerzan S., Florian R., Schafer C. and Wicentowski R. *The johns hopkins senseval2 system descriptions*. In *Proceedings of Senseval-2: Second International Workshop on Evaluating Word Sense Disambiguation Systems*. Toulouse, France, 2001.

Zelaia A., Alegria I., Arregi O. and Sierra B. *Analyzing the Effect of Dimensionality Reduction in Document Categorization for Basque*. In *Proceedings of the 2nd Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics. (H&T'05)*. Poznan, Poland, 2005.

Zelaia A., Arregi O. and Sierra B. *A multiclassifier based approach for word sense disambiguation using singular value decomposition*. In *Proceedings of the Eighth International Workshop on Computational Semantics (IWCS-08)*. Tilburg, The Netherlands, 2008.

Zelikovitz S. and Hirsh H. *Using LSI for text classification in the presence of background text*. In H. Paques, L. Liu and D. Grossman, eds., *Proceedings of CIKM-01, 10th ACM International Conference on Information and Knowledge Management*, pp. 113–118. ACM Press, New York, US, Atlanta, US, 2001.

Zhong Z., Ng H.T. and Chan Y.S. *Word sense disambiguation using OntoNotes: An empirical study*. In *Proceedings of the 2008 Conference* *on Empirical Methods in Natural Language Processing*, pp. 1002–1010. Association for Computational Linguistics, Honolulu, Hawaii, 2008.

Zhu J. and Hovy E. *Active learning for word sense disambiguation with methods for addressing the class imbalance problem*. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 783–790. Prague, Czech Republic, 2007.

Zhu X. *Semi-supervised learning literature survey*. Technical Report 1530, University of Wisconsin-Madison, 2005.

APPENDIX A

Word sense distribution from Domain-Specific Sussex Corpus

BNC dataset sense distribution

```
bank.n => 1 ---> 67 - 70.5263157894737 %
bank.n => 2 ---> 12 - 12.6315789473684 %
bank.n => 7 ---> 6 - 6.31578947368421 %
bank.n => 5 ---> 4 - 4.21052631578947 %
bank.n => 4 ---> 3 - 3.15789473684211 %
bank.n => unclear ---> 2 - 2.10526315789474 %
bank.n => 3 ---> 1 - 1.05263157894737 %
total of examples = 95
total of senses = 7

bill.n => 1 ---> 40 - 40.4040404040404 %
bill.n => 2 ---> 39 - 39.3939393939394 %
bill.n => 7 ---> 11 - 11.1111111111111 %
bill.n => unclear ---> 3 - 3.03030303030303 %
bill.n => 4 ---> 2 - 2.02020202020202 %
bill.n => unlisted-sense ---> 2 - 2.02020202020202 %
bill.n => 3 ---> 1 - 1.01010101010101 %
bill.n => 5 ---> 1 - 1.01010101010101 %
```

```
total of examples = 99
total of senses = 8

bond.n => 2 ---> 43 - 45.7446808510638 %
bond.n => 3 ---> 31 - 32.9787234042553 %
bond.n => 1 ---> 15 - 15.9574468085106 %
bond.n => 6 ---> 2 - 2.12765957446809 %
bond.n => unclear ---> 2 - 2.12765957446809 %
bond.n => 10 ---> 1 - 1.06382978723404 %
total of examples = 94
total of senses = 6

check.n => 6 ---> 11 - 33.3333333333333 %
check.n => 10 ---> 8 - 24.2424242424242 %
check.n => 2 ---> 5 - 15.1515151515152 %
check.n => 4 ---> 3 - 9.09090909090909 %
check.n => 11 ---> 2 - 6.06060606060606 %
check.n => 8 ---> 1 - 3.03030303030303 %
check.n => unlisted-sense ---> 1 - 3.03030303030303 %
check.n => 7 ---> 1 - 3.03030303030303 %
check.n => 5 ---> 1 - 3.03030303030303 %
total of examples = 33
total of senses = 9

chip.n => 7 ---> 72 - 78.2608695652174 %
chip.n => 4 ---> 9 - 9.78260869565217 %
chip.n => 1 ---> 5 - 5.43478260869565 %
chip.n => unlisted-sense ---> 3 - 3.26086956521739 %
chip.n => unclear ---> 2 - 2.17391304347826 %
chip.n => 8 ---> 1 - 1.08695652173913 %
total of examples = 92
total of senses = 6

club.n => 2 ---> 55 - 59.1397849462366 %
club.n => unclear ---> 13 - 13.9784946236559 %
club.n => 7 ---> 9 - 9.67741935483871 %
club.n => 1 ---> 8 - 8.60215053763441 %
club.n => 4 ---> 4 - 4.3010752688172 %
club.n => 5 ---> 3 - 3.2258064516129 %
club.n => unlisted-sense ---> 1 - 1.0752688172043 %
total of examples = 93
total of senses = 7

coach.n => 1 ---> 43 - 45.7446808510638 %
coach.n => 5 ---> 27 - 28.7234042553192 %
coach.n => 4 ---> 12 - 12.7659574468085 %
coach.n => unclear ---> 5 - 5.31914893617021 %
coach.n => 3 ---> 4 - 4.25531914893617 %
coach.n => 2 ---> 3 - 3.19148936170213 %
total of examples = 94
total of senses = 6

competition.n => 1 ---> 37 - 39.7849462365591 %
competition.n => 2 ---> 34 - 36.5591397849462 %
competition.n => 3 ---> 13 - 13.9784946236559 %
competition.n => unclear ---> 5 - 5.37634408602151 %
competition.n => 4 ---> 4 - 4.3010752688172 %
total of examples = 93
total of senses = 5

conversion.n => 9 ---> 42 - 47.7272727272727 %
conversion.n => 4 ---> 11 - 12.5 %
conversion.n => 6 ---> 9 - 10.2272727272727 %
conversion.n => 1 ---> 7 - 7.95454545454545 %
conversion.n => unclear ---> 6 - 6.81818181818182 %
conversion.n => 2 ---> 4 - 4.54545454545455 %
conversion.n => 3 ---> 3 - 3.40909090909091 %
conversion.n => 8 ---> 3 - 3.40909090909091 %
conversion.n => unlisted-sense ---> 3 - 3.40909090909091 %
total of examples = 88
total of senses = 9

country.n => 2 ---> 42 - 43.75 %
country.n => 1 ---> 30 - 31.25 %
country.n => 4 ---> 18 - 18.75 %
country.n => 3 ---> 2 - 2.08333333333333 %
country.n => unclear ---> 2 - 2.08333333333333 %
country.n => unlisted-sense ---> 1 - 1.04166666666667 %
country.n => 5 ---> 1 - 1.04166666666667 %
total of examples = 96
total of senses = 7

crew.n => 1 ---> 48 - 51.6129032258064 %
crew.n => 2 ---> 19 - 20.4301075268817 %
crew.n => unclear ---> 12 - 12.9032258064516 %
crew.n => 4 ---> 8 - 8.60215053763441 %
crew.n => 3 ---> 4 - 4.3010752688172 %
crew.n => unlisted-sense ---> 2 - 2.1505376344086 %
total of examples = 93
total of senses = 6

delivery.n => 1 ---> 71 - 73.9583333333333 %
delivery.n => 2 ---> 9 - 9.375 %
delivery.n => 3 ---> 6 - 6.25 %
delivery.n => unclear ---> 3 - 3.125 %
delivery.n => 5 ---> 3 - 3.125 %
delivery.n => 6 ---> 2 - 2.08333333333333 %
delivery.n => 4 ---> 2 - 2.08333333333333 %
total of examples = 96
total of senses = 7

division.n => 2 ---> 25 - 33.3333333333333 %
division.n => 7 ---> 14 - 18.6666666666667 %
division.n => 3 ---> 13 - 17.3333333333333 %
division.n => 4 ---> 7 - 9.33333333333333 %
division.n => 6 ---> 5 - 6.66666666666667 %
division.n => 12 ---> 4 - 5.33333333333333 %
division.n => unclear ---> 2 - 2.66666666666667 %
division.n => 8 ---> 2 - 2.66666666666667 %
division.n => 1 ---> 2 - 2.66666666666667 %
division.n => 5 ---> 1 - 1.33333333333333 %
total of examples = 75
total of senses = 10
```

```
fan.n => 3 ---> 39 - 40.625 %
fan.n => 2 ---> 27 - 28.125 %
fan.n => 1 ---> 17 - 17.7083333333333 %
fan.n => unlisted-sense ---> 13 - 13.5416666666667 %
total of examples = 96
total of senses = 4

fishing.n => 1 ---> 53 - 60.2272727272727 %
fishing.n => 2 ---> 27 - 30.6818181818182 %
fishing.n => unclear ---> 6 - 6.81818181818182 %
fishing.n => unlisted-sense ---> 2 - 2.27272727272727 %
total of examples = 88
total of senses = 4

goal.n => 2 ---> 46 - 46.4646464646465 %
goal.n => 1 ---> 45 - 45.4545454545455 %
goal.n => 3 ---> 5 - 5.05050505050505 %
goal.n => 4 ---> 2 - 2.02020202020202 %
goal.n => unclear ---> 1 - 1.01010101010101 %
total of examples = 99
total of senses = 5

half.n => 1 ---> 77 - 81.0526315789474 %
half.n => 2 ---> 15 - 15.7894736842105 %
half.n => unlisted-sense ---> 2 - 2.10526315789474 %
half.n => unclear ---> 1 - 1.05263157894737 %
total of examples = 95
total of senses = 4

level.n => 1 ---> 42 - 54.5454545454545 %
level.n => 4 ---> 12 - 15.5844155844156 %
level.n => 2 ---> 9 - 11.6883116883117 %
level.n => 3 ---> 8 - 10.3896103896104 %
level.n => 8 ---> 4 - 5.19480519480519 %
level.n => unclear ---> 2 - 2.5974025974026 %
total of examples = 77
total of senses = 6
```

```
manager.n => 1 ---> 71 - 71.7171717171717 %
manager.n => 2 ---> 26 - 26.2626262626263 %
manager.n => unlisted-sense ---> 1 - 1.01010101010101 %
manager.n => unclear ---> 1 - 1.01010101010101 %
total of examples = 99
total of senses = 4

market.n => 1 ---> 38 - 61.2903225806452 %
market.n => 3 ---> 12 - 19.3548387096774 %
market.n => 4 ---> 7 - 11.2903225806452 %
market.n => 2 ---> 4 - 6.45161290322581 %
market.n => unclear ---> 1 - 1.61290322580645 %
total of examples = 62
total of senses = 5

package.n => 1 ---> 43 - 48.8636363636364 %
package.n => 3 ---> 32 - 36.3636363636364 %
package.n => 2 ---> 11 - 12.5 %
package.n => unclear ---> 2 - 2.27272727272727 %
total of examples = 88
total of senses = 4

performance.n => 4 ---> 18 - 20.6896551724138 %
performance.n => 5 ---> 18 - 20.6896551724138 %
performance.n => 2 ---> 17 - 19.5402298850575 %
performance.n => 3 ---> 12 - 13.7931034482759 %
performance.n => 1 ---> 11 - 12.6436781609195 %
performance.n => unclear ---> 11 - 12.6436781609195 %
total of examples = 87
total of senses = 6

phase.n => 2 ---> 69 - 75 %
phase.n => 1 ---> 8 - 8.69565217391304 %
phase.n => unclear ---> 8 - 8.69565217391304 %
phase.n => unlisted-sense ---> 3 - 3.26086956521739 %
phase.n => 3 ---> 3 - 3.26086956521739 %
phase.n => 4 ---> 1 - 1.08695652173913 %
total of examples = 92
total of senses = 6

pitch.n => unlisted-sense ---> 41 - 63.0769230769231 %
pitch.n => 1 ---> 12 - 18.4615384615385 %
pitch.n => 5 ---> 4 - 6.15384615384615 %
pitch.n => 6 ---> 2 - 3.07692307692308 %
pitch.n => 4 ---> 2 - 3.07692307692308 %
pitch.n => unclear ---> 2 - 3.07692307692308 %
pitch.n => 2 ---> 2 - 3.07692307692308 %
total of examples = 65
total of senses = 7

receiver.n => 3 ---> 45 - 47.3684210526316 %
receiver.n => 2 ---> 22 - 23.1578947368421 %
receiver.n => 4 ---> 14 - 14.7368421052632 %
receiver.n => 1 ---> 14 - 14.7368421052632 %
total of examples = 95
total of senses = 4

record.n => 3 ---> 27 - 33.75 %
record.n => 5 ---> 15 - 18.75 %
record.n => 4 ---> 14 - 17.5 %
record.n => 1 ---> 11 - 13.75 %
record.n => 7 ---> 5 - 6.25 %
record.n => unclear ---> 5 - 6.25 %
record.n => 6 ---> 2 - 2.5 %
record.n => 8 ---> 1 - 1.25 %
total of examples = 80
total of senses = 8

reserve.n => 5 ---> 40 - 48.780487804878 %
reserve.n => 2 ---> 21 - 25.609756097561 %
reserve.n => 6 ---> 5 - 6.09756097560976 %
reserve.n => 1 ---> 5 - 6.09756097560976 %
reserve.n => unclear ---> 5 - 6.09756097560976 %
reserve.n => 7 ---> 3 - 3.65853658536585 %
reserve.n => 3 ---> 2 - 2.4390243902439 %
reserve.n => unlisted-sense ---> 1 - 1.21951219512195 %
```

```
total of examples = 82
total of senses = 8

return.n => 5 ---> 22 - 27.5 %
return.n => 6 ---> 20 - 25 %
return.n => 2 ---> 20 - 25 %
return.n => unclear ---> 5 - 6.25 %
return.n => 4 ---> 4 - 5 %
return.n => 10 ---> 4 - 5 %
return.n => 7 ---> 3 - 3.75 %
return.n => 9 ---> 1 - 1.25 %
return.n => 13 ---> 1 - 1.25 %
total of examples = 80
total of senses = 9

right.n => 3 ---> 34 - 38.6363636363636 %
right.n => 1 ---> 32 - 36.3636363636364 %
right.n => 5 ---> 10 - 11.3636363636364 %
right.n => 6 ---> 7 - 7.95454545454545 %
right.n => unlisted-sense ---> 2 - 2.27272727272727 %
right.n => 2 ---> 2 - 2.27272727272727 %
right.n => 7 ---> 1 - 1.13636363636364 %
total of examples = 88
total of senses = 7

running.n => 4 ---> 44 - 60.2739726027397 %
running.n => 3 ---> 11 - 15.0684931506849 %
running.n => 2 ---> 10 - 13.6986301369863 %
running.n => unclear ---> 3 - 4.10958904109589 %
running.n => 5 ---> 3 - 4.10958904109589 %
running.n => unlisted-sense ---> 2 - 2.73972602739726 %
total of examples = 73
total of senses = 6

score.n => 3 ---> 33 - 37.0786516853933 %
score.n => 2 ---> 20 - 22.4719101123595 %
score.n => 1 ---> 15 - 16.8539325842697 %
score.n => 4 ---> 5 - 5.61797752808989 %
score.n => 10 ---> 5 - 5.61797752808989 %
score.n => 6 ---> 3 - 3.37078651685393 %
score.n => unclear ---> 3 - 3.37078651685393 %
score.n => 5 ---> 3 - 3.37078651685393 %
score.n => 9 ---> 1 - 1.12359550561798 %
score.n => unlisted-sense ---> 1 - 1.12359550561798 %
total of examples = 89
total of senses = 10

share.n => 1 ---> 56 - 61.5384615384615 %
share.n => 2 ---> 24 - 26.3736263736264 %
share.n => 3 ---> 9 - 9.89010989010989 %
share.n => unclear ---> 2 - 2.1978021978022 %
total of examples = 91
total of senses = 4

star.n => 6 ---> 40 - 43.010752688172 %
star.n => 5 ---> 16 - 17.2043010752688 %
star.n => 3 ---> 11 - 11.8279569892473 %
star.n => unclear ---> 9 - 9.67741935483871 %
star.n => 2 ---> 6 - 6.45161290322581 %
star.n => 4 ---> 6 - 6.45161290322581 %
star.n => 1 ---> 4 - 4.3010752688172 %
star.n => 7 ---> 1 - 1.0752688172043 %
total of examples = 93
total of senses = 8

strike.n => 1 ---> 72 - 83.7209302325581 %
strike.n => unlisted-sense ---> 7 - 8.13953488372093 %
strike.n => 2 ---> 4 - 4.65116279069767 %
strike.n => unclear ---> 2 - 2.32558139534884 %
strike.n => 4 ---> 1 - 1.16279069767442 %
total of examples = 86
total of senses = 5

striker.n => 1 ---> 94 - 94 %
striker.n => 5 ---> 3 - 3 %
striker.n => 4 ---> 2 - 2 %
striker.n => 3 ---> 1 - 1 %
total of examples = 100
total of senses = 4

target.n => 5 ---> 45 - 51.7241379310345 %
target.n => unclear ---> 14 - 16.0919540229885 %
target.n => 1 ---> 11 - 12.6436781609195 %
target.n => 2 ---> 10 - 11.4942528735632 %
target.n => 3 ---> 4 - 4.59770114942529 %
target.n => 4 ---> 3 - 3.44827586206897 %
total of examples = 87
total of senses = 6

tie.n => 1 ---> 37 - 40.2173913043478 %
tie.n => 2 ---> 36 - 39.1304347826087 %
tie.n => unlisted-sense ---> 8 - 8.69565217391304 %
tie.n => 3 ---> 6 - 6.52173913043478 %
tie.n => 6 ---> 2 - 2.17391304347826 %
tie.n => unclear ---> 2 - 2.17391304347826 %
tie.n => 5 ---> 1 - 1.08695652173913 %
total of examples = 92
total of senses = 7

title.n => 4 ---> 29 - 48.3333333333333 %
title.n => 2 ---> 19 - 31.6666666666667 %
title.n => 6 ---> 8 - 13.3333333333333 %
title.n => unlisted-sense ---> 1 - 1.66666666666667 %
title.n => 3 ---> 1 - 1.66666666666667 %
title.n => 7 ---> 1 - 1.66666666666667 %
title.n => unclear ---> 1 - 1.66666666666667 %
total of examples = 60
total of senses = 7

top.n => 1 ---> 31 - 50 %
top.n => 10 ---> 9 - 14.5161290322581 %
top.n => 2 ---> 8 - 12.9032258064516 %
top.n => 3 ---> 5 - 8.06451612903226 %
top.n => 5 ---> 5 - 8.06451612903226 %
top.n => unclear ---> 2 - 3.2258064516129 %
top.n => 9 ---> 2 - 3.2258064516129 %
total of examples = 62
total of senses = 7

transfer.n => 1 ---> 32 - 41.5584415584416 %
transfer.n => 6 ---> 29 - 37.6623376623377 %
transfer.n => unclear ---> 7 - 9.09090909090909 %
transfer.n => 3 ---> 5 - 6.49350649350649 %
transfer.n => 2 ---> 4 - 5.19480519480519 %
total of examples = 77
total of senses = 5

will.n => 2 ---> 23 - 44.2307692307692 %
will.n => 3 ---> 20 - 38.4615384615385 %
will.n => 1 ---> 6 - 11.5384615384615 %
will.n => unclear ---> 3 - 5.76923076923077 %
total of examples = 52
total of senses = 4
```

Sports dataset sense distribution

```
bank.n => 1 ---> 68 - 87.1794871794872 %
bank.n => 7 ---> 6 - 7.69230769230769 %
bank.n => 2 ---> 3 - 3.84615384615385 %
bank.n => 4 ---> 1 - 1.28205128205128 %
total of examples = 78
total of senses = 5

bill.n => 2 ---> 28 - 44.4444444444444 %
bill.n => 1 ---> 22 - 34.9206349206349 %
bill.n => 4 ---> 11 - 17.4603174603175 %
bill.n => unlisted-sense ---> 1 - 1.58730158730159 %
bill.n => 3 ---> 1 - 1.58730158730159 %
total of examples = 63
total of senses = 6
```

```
bond.n => 2 ---> 39 - 73.5849056603774 %
bond.n => 3 ---> 7 - 13.2075471698113 %
bond.n => 4 ---> 6 - 11.3207547169811 %
bond.n => unclear ---> 1 - 1.88679245283019 %
total of examples = 53
total of senses = 5

check.n => 1 ---> 27 - 50 %
check.n => 4 ---> 9 - 16.6666666666667 %
check.n => 12 ---> 9 - 16.6666666666667 %
check.n => 6 ---> 7 - 12.962962962963 %
check.n => 11 ---> 2 - 3.7037037037037 %
total of examples = 54
total of senses = 6

chip.n => 8 ---> 75 - 87.2093023255814 %
chip.n => 4 ---> 4 - 4.65116279069767 %
chip.n => unlisted-sense ---> 4 - 4.65116279069767 %
chip.n => 6 ---> 1 - 1.16279069767442 %
chip.n => 1 ---> 1 - 1.16279069767442 %
chip.n => 7 ---> 1 - 1.16279069767442 %
total of examples = 86
total of senses = 7

club.n => 2 ---> 78 - 82.9787234042553 %
club.n => 1 ---> 8 - 8.51063829787234 %
club.n => 5 ---> 5 - 5.31914893617021 %
club.n => 4 ---> 2 - 2.12765957446809 %
club.n => unclear ---> 1 - 1.06382978723404 %
total of examples = 94
total of senses = 6

coach.n => 1 ---> 94 - 95.9183673469388 %
coach.n => unclear ---> 2 - 2.04081632653061 %
coach.n => 2 ---> 2 - 2.04081632653061 %
total of examples = 98
total of senses = 4
```

```
competition.n => 2 ---> 92 - 94.8453608247423 %
competition.n => 3 ---> 3 - 3.09278350515464 %
competition.n => 1 ---> 1 - 1.03092783505155 %
competition.n => unclear ---> 1 - 1.03092783505155 %
total of examples = 97
total of senses = 5

conversion.n => 3 ---> 100 - 100 %
total of examples = 100
total of senses = 2

country.n => 2 ---> 62 - 72.9411764705882 %
country.n => 1 ---> 15 - 17.6470588235294 %
country.n => 3 ---> 7 - 8.23529411764706 %
country.n => unlisted-sense ---> 1 - 1.17647058823529 %
total of examples = 85
total of senses = 5

crew.n => 4 ---> 70 - 79.5454545454545 %
crew.n => 2 ---> 8 - 9.09090909090909 %
crew.n => unclear ---> 7 - 7.95454545454545 %
crew.n => 1 ---> 3 - 3.40909090909091 %
total of examples = 88
total of senses = 5

delivery.n => 6 ---> 97 - 97 %
delivery.n => 1 ---> 2 - 2 %
delivery.n => unclear ---> 1 - 1 %
total of examples = 100
total of senses = 4

division.n => 7 ---> 69 - 98.5714285714286 %
division.n => unclear ---> 1 - 1.42857142857143 %
total of examples = 70
total of senses = 3

fan.n => 2 ---> 94 - 95.9183673469388 %
fan.n => 3 ---> 4 - 4.08163265306122 %
```

```
total of examples = 98
total of senses = 3

fishing.n => 1 ---> 32 - 91.4285714285714 %
fishing.n => 2 ---> 3 - 8.57142857142857 %
total of examples = 35
total of senses = 3

goal.n => 2 ---> 90 - 90 %
goal.n => 1 ---> 4 - 4 %
goal.n => 3 ---> 4 - 4 %
goal.n => unclear ---> 2 - 2 %
total of examples = 100
total of senses = 5

half.n => 2 ---> 69 - 74.1935483870968 %
half.n => 1 ---> 22 - 23.6559139784946 %
half.n => unlisted-sense ---> 1 - 1.0752688172043 %
half.n => unclear ---> 1 - 1.0752688172043 %
total of examples = 93
total of senses = 5

level.n => unlisted-sense ---> 35 - 42.6829268292683 %
level.n => 1 ---> 14 - 17.0731707317073 %
level.n => 2 ---> 13 - 15.8536585365854 %
level.n => 4 ---> 12 - 14.6341463414634 %
level.n => 3 ---> 5 - 6.09756097560976 %
level.n => unclear ---> 2 - 2.4390243902439 %
level.n => 7 ---> 1 - 1.21951219512195 %
total of examples = 82
total of senses = 8

manager.n => 2 ---> 86 - 90.5263157894737 %
manager.n => 1 ---> 8 - 8.42105263157895 %
manager.n => unclear ---> 1 - 1.05263157894737 %
total of examples = 95
total of senses = 4
```

```
market.n => 2 ---> 35 - 43.75 %
market.n => 1 ---> 32 - 40 %
market.n => 3 ---> 6 - 7.5 %
market.n => unlisted-sense ---> 3 - 3.75 %
market.n => 4 ---> 2 - 2.5 %
market.n => unclear ---> 2 - 2.5 %
total of examples = 80
total of senses = 7

package.n => 1 ---> 87 - 93.5483870967742 %
package.n => 2 ---> 5 - 5.37634408602151 %
package.n => unclear ---> 1 - 1.0752688172043 %
total of examples = 93
total of senses = 4

performance.n => 5 ---> 80 - 84.2105263157895 %
performance.n => unclear ---> 8 - 8.42105263157895 %
performance.n => 2 ---> 4 - 4.21052631578947 %
performance.n => 4 ---> 2 - 2.10526315789474 %
performance.n => 1 ---> 1 - 1.05263157894737 %
total of examples = 95
total of senses = 6

phase.n => 2 ---> 99 - 99 %
phase.n => unclear ---> 1 - 1 %
total of examples = 100
total of senses = 3

pitch.n => unlisted-sense ---> 59 - 70.2380952380952 %
pitch.n => 2 ---> 22 - 26.1904761904762 %
pitch.n => 7 ---> 3 - 3.57142857142857 %
total of examples = 84
total of senses = 4

receiver.n => 5 ---> 92 - 92 %
receiver.n => 2 ---> 5 - 5 %
receiver.n => 1 ---> 3 - 3 %
total of examples = 100
total of senses = 4

record.n => 3 ---> 63 - 68.4782608695652 %
record.n => 2 ---> 16 - 17.3913043478261 %
record.n => 5 ---> 12 - 13.0434782608696 %
record.n => 7 ---> 1 - 1.08695652173913 %
total of examples = 92
total of senses = 5

reserve.n => 3 ---> 75 - 82.4175824175824 %
reserve.n => 2 ---> 8 - 8.79120879120879 %
reserve.n => unclear ---> 5 - 5.49450549450549 %
reserve.n => 5 ---> 2 - 2.1978021978022 %
reserve.n => 6 ---> 1 - 1.0989010989011 %
total of examples = 91
total of senses = 6

return.n => 5 ---> 23 - 29.4871794871795 %
return.n => 2 ---> 19 - 24.3589743589744 %
return.n => 11 ---> 12 - 15.3846153846154 %
return.n => 12 ---> 9 - 11.5384615384615 %
return.n => 13 ---> 6 - 7.69230769230769 %
return.n => 10 ---> 4 - 5.12820512820513 %
return.n => unclear ---> 2 - 2.56410256410256 %
return.n => 6 ---> 1 - 1.28205128205128 %
return.n => 4 ---> 1 - 1.28205128205128 %
return.n => unlisted-sense ---> 1 - 1.28205128205128 %
total of examples = 78
total of senses = 11

right.n => 3 ---> 38 - 58.4615384615385 %
right.n => 1 ---> 18 - 27.6923076923077 %
right.n => 7 ---> 5 - 7.69230769230769 %
right.n => 6 ---> 2 - 3.07692307692308 %
right.n => unclear ---> 2 - 3.07692307692308 %
total of examples = 65
total of senses = 6
```

```
running.n => unlisted-sense ---> 20 - 25.3164556962025 %
running.n => 5 ---> 17 - 21.5189873417722 %
running.n => 2 ---> 15 - 18.9873417721519 %
running.n => 1 ---> 11 - 13.9240506329114 %
running.n => 4 ---> 9 - 11.3924050632911 %
running.n => 3 ---> 4 - 5.06329113924051 %
running.n => unclear ---> 3 - 3.79746835443038 %
total of examples = 79
total of senses = 8

score.n => 3 ---> 69 - 83.1325301204819 %
score.n => 10 ---> 12 - 14.4578313253012 %
score.n => 4 ---> 1 - 1.20481927710843 %
score.n => unclear ---> 1 - 1.20481927710843 %
total of examples = 83
total of senses = 5

share.n => 3 ---> 46 - 47.9166666666667 %
share.n => 1 ---> 38 - 39.5833333333333 %
share.n => 2 ---> 12 - 12.5 %
total of examples = 96
total of senses = 4

star.n => 2 ---> 74 - 81.3186813186813 %
star.n => 6 ---> 14 - 15.3846153846154 %
star.n => 4 ---> 2 - 2.1978021978022 %
star.n => unclear ---> 1 - 1.0989010989011 %
total of examples = 91
total of senses = 5

strike.n => unlisted-sense ---> 54 - 59.3406593406593 %
strike.n => 1 ---> 24 - 26.3736263736264 %
strike.n => 3 ---> 11 - 12.0879120879121 %
strike.n => 4 ---> 1 - 1.0989010989011 %
strike.n => unclear ---> 1 - 1.0989010989011 %
total of examples = 91
total of senses = 6
```

```
striker.n => 1 ---> 97 - 97 %
striker.n => unclear ---> 3 - 3 %
total of examples = 100
total of senses = 3

target.n => 5 ---> 70 - 81.3953488372093 %
target.n => 2 ---> 10 - 11.6279069767442 %
target.n => unclear ---> 4 - 4.65116279069767 %
target.n => 1 ---> 2 - 2.32558139534884 %
total of examples = 86
total of senses = 5

tie.n => unlisted-sense ---> 34 - 40.4761904761905 %
tie.n => 6 ---> 25 - 29.7619047619048 %
tie.n => 3 ---> 22 - 26.1904761904762 %
tie.n => unclear ---> 2 - 2.38095238095238 %
tie.n => 1 ---> 1 - 1.19047619047619 %
total of examples = 84
total of senses = 6

title.n => 4 ---> 98 - 100 %
total of examples = 98
total of senses = 2

top.n => 5 ---> 56 - 96.551724137931 %
top.n => 1 ---> 2 - 3.44827586206897 %
total of examples = 58
total of senses = 3

transfer.n => 6 ---> 74 - 91.358024691358 %
transfer.n => 2 ---> 5 - 6.17283950617284 %
transfer.n => 1 ---> 1 - 1.23456790123457 %
transfer.n => unclear ---> 1 - 1.23456790123457 %
total of examples = 81
total of senses = 5

will.n => 2 ---> 23 - 62.1621621621622 %
will.n => 1 ---> 13 - 35.1351351351351 %
will.n => 3 ---> 1 - 2.7027027027027 %
total of examples = 37
total of senses = 4
```

Finance dataset sense distribution

```
bank.n => 1 ---> 95 - 100 %
total of examples = 95
total of senses = 2

bill.n => 1 ---> 67 - 75.2808988764045 %
bill.n => 2 ---> 16 - 17.9775280898876 %
bill.n => 3 ---> 4 - 4.49438202247191 %
bill.n => unclear ---> 2 - 2.24719101123596 %
total of examples = 89
total of senses = 5

bond.n => 2 ---> 96 - 96.969696969697 %
bond.n => unclear ---> 3 - 3.03030303030303 %
total of examples = 99
total of senses = 3

check.n => 1 ---> 19 - 44.1860465116279 %
check.n => 4 ---> 15 - 34.8837209302326 %
check.n => 6 ---> 9 - 20.9302325581395 %
total of examples = 43
total of senses = 4

chip.n => 7 ---> 51 - 77.2727272727273 %
chip.n => unclear ---> 9 - 13.6363636363636 %
chip.n => 6 ---> 3 - 4.54545454545455 %
chip.n => unlisted-sense ---> 2 - 3.03030303030303 %
chip.n => 9 ---> 1 - 1.51515151515152 %
total of examples = 66
total of senses = 6

club.n => 2 ---> 92 - 93.8775510204082 %
club.n => 3 ---> 2 - 2.04081632653061 %
club.n => unclear ---> 2 - 2.04081632653061 %
club.n => 4 ---> 1 - 1.02040816326531 %
club.n => 7 ---> 1 - 1.02040816326531 %
total of examples = 98
total of senses = 6

coach.n => 5 ---> 35 - 61.4035087719298 %
coach.n => 1 ---> 12 - 21.0526315789474 %
coach.n => 3 ---> 7 - 12.280701754386 %
coach.n => 4 ---> 2 - 3.50877192982456 %
coach.n => unclear ---> 1 - 1.75438596491228 %
total of examples = 57
total of senses = 6

competition.n => 1 ---> 90 - 93.75 %
competition.n => 4 ---> 2 - 2.08333333333333 %
competition.n => unclear ---> 2 - 2.08333333333333 %
competition.n => 3 ---> 1 - 1.04166666666667 %
competition.n => 2 ---> 1 - 1.04166666666667 %
total of examples = 96
total of senses = 6

conversion.n => 8 ---> 62 - 70.4545454545455 %
conversion.n => 9 ---> 14 - 15.9090909090909 %
conversion.n => unclear ---> 6 - 6.81818181818182 %
conversion.n => 6 ---> 4 - 4.54545454545455 %
conversion.n => 3 ---> 1 - 1.13636363636364 %
conversion.n => 2 ---> 1 - 1.13636363636364 %
total of examples = 88
total of senses = 7

country.n => 2 ---> 91 - 92.8571428571429 %
country.n => 1 ---> 5 - 5.10204081632653 %
country.n => 4 ---> 1 - 1.02040816326531 %
country.n => 3 ---> 1 - 1.02040816326531 %
total of examples = 98
total of senses = 5
```

```
crew.n => 1 ---> 81 - 84.375 %
crew.n => 2 ---> 9 - 9.375 %
crew.n => unclear ---> 5 - 5.20833333333333 %
crew.n => 4 ---> 1 - 1.04166666666667 %
total of examples = 96
total of senses = 5

delivery.n => unclear ---> 70 - 72.1649484536082 %
delivery.n => 1 ---> 20 - 20.6185567010309 %
delivery.n => 3 ---> 5 - 5.15463917525773 %
delivery.n => 4 ---> 2 - 2.06185567010309 %
total of examples = 97
total of senses = 5

division.n => 2 ---> 70 - 76.0869565217391 %
division.n => 6 ---> 11 - 11.9565217391304 %
division.n => 4 ---> 5 - 5.43478260869565 %
division.n => 3 ---> 4 - 4.34782608695652 %
division.n => 7 ---> 1 - 1.08695652173913 %
division.n => unclear ---> 1 - 1.08695652173913 %
total of examples = 92
total of senses = 7

fan.n => 3 ---> 15 - 39.4736842105263 %
fan.n => 2 ---> 12 - 31.5789473684211 %
fan.n => 1 ---> 11 - 28.9473684210526 %
total of examples = 38
total of senses = 4

fishing.n => 2 ---> 81 - 89.010989010989 %
fishing.n => 1 ---> 10 - 10.989010989011 %
total of examples = 91
total of senses = 3

goal.n => 1 ---> 100 - 100 %
total of examples = 100
total of senses = 2
```

```
half.n => 1 ---> 99 - 100 %
total of examples = 99
total of senses = 2

level.n => 1 ---> 75 - 85.2272727272727 %
level.n => 3 ---> 6 - 6.81818181818182 %
level.n => unclear ---> 5 - 5.68181818181818 %
level.n => 2 ---> 2 - 2.27272727272727 %
total of examples = 88
total of senses = 5

manager.n => 1 ---> 91 - 94.7916666666667 %
manager.n => 2 ---> 4 - 4.16666666666667 %
manager.n => unclear ---> 1 - 1.04166666666667 %
total of examples = 96
total of senses = 4

market.n => 2 ---> 52 - 70.2702702702703 %
market.n => 1 ---> 19 - 25.6756756756757 %
market.n => 3 ---> 3 - 4.05405405405405 %
total of examples = 74
total of senses = 4

package.n => 1 ---> 90 - 91.8367346938776 %
package.n => 2 ---> 7 - 7.14285714285714 %
package.n => 3 ---> 1 - 1.02040816326531 %
total of examples = 98
total of senses = 4

performance.n => 2 ---> 64 - 87.6712328767123 %
performance.n => 5 ---> 4 - 5.47945205479452 %
performance.n => 4 ---> 2 - 2.73972602739726 %
performance.n => unclear ---> 2 - 2.73972602739726 %
performance.n => 1 ---> 1 - 1.36986301369863 %
total of examples = 73
total of senses = 6
```

```
phase.n => 2 ---> 82 - 97.6190476190476 %
phase.n => unlisted-sense ---> 1 - 1.19047619047619 %
phase.n => unclear ---> 1 - 1.19047619047619 %
total of examples = 84
total of senses = 4

pitch.n => 4 ---> 37 - 84.0909090909091 %
pitch.n => 2 ---> 4 - 9.09090909090909 %
pitch.n => unlisted-sense ---> 3 - 6.81818181818182 %
total of examples = 44
total of senses = 4

receiver.n => 2 ---> 84 - 89.3617021276596 %
receiver.n => 4 ---> 6 - 6.38297872340426 %
receiver.n => 3 ---> 2 - 2.12765957446809 %
receiver.n => 1 ---> 1 - 1.06382978723404 %
receiver.n => 5 ---> 1 - 1.06382978723404 %
total of examples = 94
total of senses = 6

record.n => 3 ---> 80 - 80.8080808080808 %
record.n => 5 ---> 14 - 14.1414141414141 %
record.n => 7 ---> 3 - 3.03030303030303 %
record.n => 4 ---> 1 - 1.01010101010101 %
record.n => unclear ---> 1 - 1.01010101010101 %
total of examples = 99
total of senses = 6

reserve.n => 2 ---> 71 - 98.6111111111111 %
reserve.n => unclear ---> 1 - 1.38888888888889 %
total of examples = 72
total of senses = 3

return.n => 6 ---> 32 - 40 %
return.n => 10 ---> 17 - 21.25 %
return.n => 5 ---> 12 - 15 %
return.n => 2 ---> 8 - 10 %
return.n => 1 ---> 4 - 5 %
return.n => 4 ---> 4 - 5 %
return.n => unclear ---> 2 - 2.5 %
return.n => 13 ---> 1 - 1.25 %
total of examples = 80
total of senses = 9

right.n => 1 ---> 48 - 70.5882352941177 %
right.n => 5 ---> 15 - 22.0588235294118 %
right.n => 6 ---> 4 - 5.88235294117647 %
right.n => unlisted-sense ---> 1 - 1.47058823529412 %
total of examples = 68
total of senses = 5

running.n => 4 ---> 22 - 40 %
running.n => 3 ---> 16 - 29.0909090909091 %
running.n => unlisted-sense ---> 13 - 23.6363636363636 %
running.n => unclear ---> 3 - 5.45454545454545 %
running.n => 2 ---> 1 - 1.81818181818182 %
total of examples = 55
total of senses = 6

score.n => 4 ---> 60 - 68.1818181818182 %
score.n => 1 ---> 7 - 7.95454545454545 %
score.n => 5 ---> 6 - 6.81818181818182 %
score.n => 9 ---> 5 - 5.68181818181818 %
score.n => 3 ---> 4 - 4.54545454545455 %
score.n => 10 ---> 4 - 4.54545454545455 %
score.n => unclear ---> 1 - 1.13636363636364 %
score.n => 2 ---> 1 - 1.13636363636364 %
total of examples = 88
total of senses = 9

share.n => 1 ---> 64 - 65.3061224489796 %
share.n => 2 ---> 26 - 26.530612244898 %
share.n => 3 ---> 8 - 8.16326530612245 %
total of examples = 98
total of senses = 4
```

```
star.n => 6 ---> 38 - 40.8602150537634 %
star.n => 2 ---> 38 - 40.8602150537634 %
star.n => 3 ---> 6 - 6.45161290322581 %
star.n => 5 ---> 6 - 6.45161290322581 %
star.n => 4 ---> 4 - 4.3010752688172 %
star.n => unclear ---> 1 - 1.0752688172043 %
total of examples = 93
total of senses = 7

strike.n => 1 ---> 100 - 100 %
total of examples = 100
total of senses = 2

striker.n => 3 ---> 93 - 95.8762886597938 %
striker.n => unlisted-sense ---> 4 - 4.12371134020619 %
total of examples = 97
total of senses = 3

target.n => 5 ---> 86 - 93.4782608695652 %
target.n => 2 ---> 3 - 3.26086956521739 %
target.n => unclear ---> 2 - 2.17391304347826 %
target.n => 1 ---> 1 - 1.08695652173913 %
total of examples = 92
total of senses = 5

tie.n => 2 ---> 99 - 99 %
tie.n => 1 ---> 1 - 1 %
total of examples = 100
total of senses = 3

title.n => 6 ---> 16 - 40 %
title.n => 2 ---> 8 - 20 %
title.n => 1 ---> 8 - 20 %
title.n => 3 ---> 2 - 5 %
title.n => unclear ---> 2 - 5 %
title.n => 5 ---> 2 - 5 %
title.n => 7 ---> 1 - 2.5 %
title.n => 4 ---> 1 - 2.5 %
```

```
total of examples = 40
total of senses = 9

top.n => 5 ---> 60 - 88.2352941176471 %
top.n => unlisted-sense ---> 7 - 10.2941176470588 %
top.n => 1 ---> 1 - 1.47058823529412 %
total of examples = 68
total of senses = 4

transfer.n => 6 ---> 73 - 83.9080459770115 %
transfer.n => 1 ---> 9 - 10.3448275862069 %
transfer.n => 3 ---> 2 - 2.29885057471264 %
transfer.n => unclear ---> 1 - 1.14942528735632 %
transfer.n => 2 ---> 1 - 1.14942528735632 %
transfer.n => 5 ---> 1 - 1.14942528735632 %
total of examples = 87
total of senses = 7

will.n => 2 ---> 33 - 94.2857142857143 %
will.n => 1 ---> 2 - 5.71428571428571 %
total of examples = 35
total of senses = 3
```

APPENDIX B

Set of original features

Local collocations:

```
unigram

post_J_lem
post_J_wf
post_N_lem
post_N_wf
post_R_lem
post_R_wf
post_V_lem
post_V_wf
prev_J_lem
prev_J_wf
prev_N_lem
prev_N_wf
prev_R_lem
prev_R_wf
prev_V_lem
prev_V_wf

big_lem_cont_+1
big_lem_cont_-1
```

```
big_lem_func_+1
big_lem_func_-1
big_pos_+1
big_pos_-1
big_wf_cont_+1
big_wf_cont_-1
big_wf_func_+1
big_wf_func_-1

trig_lem_cont_+1
trig_lem_cont_-1
trig_lem_cont_0
trig_lem_func_+1
trig_lem_func_-1
trig_lem_func_0
trig_pos_+1
trig_pos_-1
trig_pos_0
trig_wf_cont_+1
trig_wf_cont_-1
trig_wf_cont_0
trig_wf_func_+1
trig_wf_func_-1
trig_wf_func_0
```

Syntactic dependencies:

```
DominatingNoun
NounModifier
Object
ObjectTo
ObjectToPreposition
Preposition
Sibling
SubjectTo
```

Bag-of-words features:

```
win_cont_lem_4w
win_cont_lem_context
pedersen_bigr
```

Domain features:

```
Domains_A
Domains_F
```