Knowledge-Lean Approaches to Metonymy Recognition

Yves Peirsman

Supervisor: Mirella Lapata

Master of Science
in
Speech and Language Processing
Theoretical and Applied Linguistics
School of Philosophy, Psychology and Language Sciences
University of Edinburgh

2005

Abstract

Current approaches to metonymy recognition are mainly supervised, relying heavily on the manual annotation of training and test data. This forms a considerable hindrance to their application on a wider scale. This dissertation therefore aims to relieve the knowledge acquisition bottleneck with respect to metonymy recognition by examining knowledge-lean approaches that reduce this need for human effort.

This investigation involves the study of three algorithms that constitute an entire spectrum of machine learning approaches — unsupervised, supervised and semi-supervised ones. Chapter 2 will discuss an unsupervised approach to metonymy recognition, and will show that promising results can be reached when the data are automatically annotated with grammatical information. Although the robustness of these systems is limited, they can serve as a pre-processing step for the selection of useful training data, thereby reducing the workload for human annotators.

Chapter 3 will investigate memory-based learning, a “lazy” supervised algorithm. This algorithm, which relies on an extremely simple learning stage, is able to replicate the results of more complex systems. Yet, it will also become clear that the performance of this algorithm, like that of others in the literature, depends heavily on grammatical annotation.

Finally, chapter 4 will present a semi-supervised algorithm that produces very promising results with only ten labelled training instances. In addition, it will be shown that less than half of the training data from chapter 3 can lead to the same performance as the entire set. Semantic information in particular will prove very useful in this respect.

In short, this dissertation presents experimental results which indicate that the knowledge acquisition bottleneck in metonymy recognition can be relieved with unsupervised and semi-supervised methods. These approaches may make the extension of current algorithms to a wide-scale metonymy resolution system a much more feasible task.

Acknowledgements

On the next page I declare that "this thesis was composed by myself". While this is doubtlessly correct, this dissertation is a composition in which the assistance of many people resounds. First of all, I would like to thank my supervisor, Mirella Lapata, whose comments and guidance always proved helpful when I found myself lost in a confusing multitude of figures or data. I could not have written this dissertation without her feedback and support.

In my exploration of the fields of Word Sense Disambiguation and metonymy resolution, I received help from a number of researchers. Katja Markert and Malvina Nissim provided me with useful information about their Mascara study, while Ted Pedersen and Anagha Kulkarni helped me use and understand their SenseClusters package. Their assistance was much appreciated.

This piece of research is the most tangible result of a year of studies in Edinburgh, an unforgettable experience that I would have had to miss if it were not for the support of a number of people. Financial support came from my parents, the Arts and Humanities Research Council, and the Royal Belgian Benevolent Society. For these last two scholarships I am indebted to my three referees — Dirk Geeraerts and Kristin Davidse at the University of Leuven, Belgium, and William McGregor at the University of Aarhus, Denmark. For support of a totally different — but equally indispensable — kind, I again thank my parents, my friends and family in Belgium, Dirk Geeraerts, and the many people who have I had the pleasure to meet here in Edinburgh — most particularly, my flatmates and fellow MSc students.

Finally, I am not a computer scientist or a native speaker of English, and my attempts at success in those fields were guided by a few close friends. I thank Dominiek Maschelein for installing Linux on my laptop, Wouter Goossens for infecting me with his enthusiasm about LaTeX, and Daniel Donavanik for proofreading an earlier draft of this dissertation.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text. This work has not been submitted for any other degree or professional qualification except as specified.

(*Yves Peirsman*)

Table of Contents

Introduction

The past decade has witnessed an upsurge of interest in metonymy, a figure of speech which uses “one entity to refer to another that is related to it” (Lakoff and Johnson, 1980, p.35). For instance, in the sentence *we are reading Shakespeare* (Kövecses and Radden, 1998, p.57), the author *Shakespeare* metonymically stands for one of his works. Cognitive as well as computational linguists seem to have realized that metonymy is a figure of speech that is extremely frequent in everyday language, and therefore constitutes an important focus of investigation. In cognitive linguistics, this investigation is concerned with accounting for the wide variety in metonymical patterns, their relationship with metaphor, etc. (see e.g. Lakoff and Johnson; Kövecses 2002). In computational linguistics, researchers study how computers can learn to recognize and interpret metonymies (see e.g. Markert and Nissim, 2002a), among other issues.

This dissertation focuses on the latter of these concerns. Computational metonymy resolution constitutes an essential part of many tasks throughout the field of Natural Language Processing. Machine translation, in particular, needs to disambiguate metonymies in the source language and determine if they can be transferred directly to the target language. For instance, while you can say in English that you *filled up the car* (where the car stands for the petrol tank), the Dutch equivalent (*de auto opvullen*) would imply that you literally filled the entire car with petrol. A translation system should be able to address such instances correctly. Interest in metonymy resolution, however, should not be limited to computational linguistics. Theoretical linguists as well can use it to study what clues are needed to disambiguate a possible metonymy, or to see if there are differences in difficulty between several metonymical patterns.

As the first chapter of this dissertation will show, research into metonymy resolution is still in its infancy. Its first systems demand much human effort and large resources. Annotators are essential for the construction of knowledge bases, or for the manual construction of training examples. This is a major weakness of the present approaches, because every new metonymical pattern that is investigated requires the construction of annotated examples. This dissertation therefore tries to determine if this human intervention can be sidestepped, or at least minimized. This question can be approached in a number of ways.

First there is the possibility of dispensing with human intervention altogether. Computational linguists have developed so-called unsupervised Word Sense Discrimination algorithms, which are aimed at the automatic discrimination of the several senses with which a word can be used. Such algorithms may thus be able to distinguish metonymical readings from literal ones. They will therefore be discussed in chapter 2.

Chapter 3 will then look at the opposite side of the spectrum and visit a supervised approach, memory-based learning. While this particular approach still requires the manual annotation of training examples, it is based on an extremely simple training algorithm that merely saves all training examples in its memory. It will become clear that this algorithm can be very successful, as long as grammatical information is taken into account. I will therefore test if this grammatical annotation can be done automatically, and at what cost.

Unsurprisingly, it will become evident the supervised approach is much more robust than its unsupervised competitor. The final chapter of this dissertation will therefore try and strike a balance between the two. It will investigate how much data we actually need, and if it is possible to develop a semi-supervised algorithm that requires only a handful of manually labelled examples.

In short, by addressing the minimization of human intervention in metonymy resolution, this dissertation presents a necessary addition to the present literature. Only if manual annotation or the construction of knowledge bases is reduced to a minimum will large-scale metonymy resolution become a feasible task.

Chapter 1

Introduction to Metonymy Resolution

As the introduction has made clear, this dissertation is to be situated against the wider backdrop of statistical metonymy resolution. This chapter therefore starts with a general introduction of metonymy in section 1.1. In section 1.2, I will discuss how the related issues of metonymy recognition and interpretation have been addressed so far in the literature. I will finish in section 1.3 with an overview of Markert and Nissim's study (see Markert and Nissim, 2002a, 2005b; Nissim and Markert, 2003, 2005), which formed the major inspiration for this dissertation.

1.1 Metonymy

Metonymy is a figure of speech that uses "one entity to refer to another that is related to it" (Lakoff and Johnson, 1980, p.35). Here are some typical examples:

(1.1) *Nixon* bombed Hanoi. (Kövecses, 2002, p.143)

(1.2) We are reading *Shakespeare*. (Kövecses and Radden, 1998, p.57)

(1.3) Tony Blair is the Prime Minister of *England*. (Peirsman and Geeraerts, acc, p.11)

(1.4) I drank a *glass* too many. (Peirsman and Geeraerts, p.12)

Despite the literal meaning of example (1.1), president Nixon did not bomb Hanoi himself. Instead, he left this to the army under his command. Similarly, *Shakespeare* in (1.2) does not refer to the writer, but to one of his works, just as *England* in (1.3) stands for the entire United Kingdom, and *glass* in (1.4) for the drink in the glass. All these examples of figurative language share one characteristic: the relationship between the two entities involved is one of contiguity (see e.g. Norrick, 1981).

In theory, this relationship of contiguity can take on countless forms (Nunberg, 1978), as is shown by the creative use of metonymy in example (1.5), where *the ham sandwich* stands for the customer that ordered it:

(1.5) The ham sandwich is waiting for his check. (Markert and Nissim, 2002a, p.1)

In practice, however, most metonymies can be classified into a restricted number of metonymical patterns (see e.g. Lakoff and Johnson, 1980; Peirsman and Geeraerts, acc). Example (1.1), for instance, belongs to the pattern `controller-for-controlled`, because Nixon stands for the army that he controls. In the same vein, example (1.2) can be classified as `artist-for-work`, (1.3) as `part-for-whole` and (1.4) as `container-for-contained`.

In addition to these *referential* metonymies, there is a second type of metonymy, which is called *logical metonymy* (see e.g. Pustejovsky, 1995). Here the polysemy arises from a mismatch between the semantics of two related words, such as a verb and its object. For instance, the interaction between the verb *finish* and its object *the cigarette* causes sentence (1.6a) to be interpreted as (1.6b):

(1.6) a. Mary finished the cigarette.

b. Mary finished smoking the cigarette. (Lapata and Lascarides, 2003, p.261)

This happens because *finish* is a verb that selects for an event as its object, whereas *the cigarette* refers to an entity. Therefore the object is type-shifted, and interpreted as “smoking the cigarette”. A computational approach to logical metonymy has already been developed by Lapata and Lascarides, and will not further concern us here.

Instead, this dissertation focuses on referential metonymies that involve location or organization names. Markert and Nissim (2005a) developed a detailed annotation scheme for metonymies of these classes, which contains a hierarchically organized inventory of the patterns they can belong to. This scheme identifies the following location patterns:

(1.7) `place-for-people`: This morning in Bonn, Dr Kohl will preside at an emergency cabinet meeting to discuss how *West Germany* should respond to the events of recent days. (British National Corpus, henceforth BNC)

(1.8) `place-for-event`: In his last assignment as Minister of State at the United Kingdom Foreign and Commonwealth Office, Francis Maude visited China on July 25-27, primarily for talks on *Hong Kong*. (BNC)

(1.9) `place-for-product`: I bought a real *Meissen* (Markert and Nissim, p.14)

In example (1.7), *West Germany* refers to the government of that country, while in (1.8), the minister talked about events concerning *Hong Kong*, not about the geographical area itself. The city in example (1.9), finally, refers to the porcelain that is made there.

Similarly, Markert and Nissim (2005a) enumerate a range of metonymical patterns that are typical of organization names:

(1.10) `organization-for-members`: *Peugeot* said that it had lost production of 49,000 cars as a result of the strikes. (BNC)

(1.11) `organization-for-product`: It was the largest *Fiat* anyone had ever seen. (BNC)

(1.12) `organization-for-facility`: If you work in *Marks and Spencers* you wear er their uniform. (BNC)

(1.13) `organization-for-index`: Cheery figures from both Tesco (up 4p at 255p) and Next (gaining 5 to 71p) put consumer-related stocks on a firm footing from the outset, with J Sainsbury adding 6 to 386p, *Kingfisher* up 2 at 463p and Argos rising 3 to 234p. (BNC)

(1.14) `organization-for-event`: “He is a guy who likes to break things up,” commented Curt Rohrman, a First Boston analyst who follows *IBM*. (BNC)

As these examples show, organizations can stand for their members (the management of Peugeot in example 1.10), for their products (*Fiat* in example 1.11), for the buildings in which they are located (example 1.12), for their shares in the stock market (example 1.13), or for events that are related to them (example 1.14). This dissertation addresses the automatic recognition of these location and organization patterns.

It is clear that metonymy is an extremely frequent phenomenon in everyday language, and many NLP tasks such as machine translation or dialogue systems have to be well-equipped to handle it correctly. They have to be able to spot a metonymy, and often also to interpret it. This combined task of recognition and interpretation is called metonymy resolution.

1.2 Metonymy resolution

Metonymy resolution involves two separate stages (Fass, 1997). The first stage, *metonymy recognition*, is meant to identify the metonymical words. The second stage, *metonymy interpretation*, then tries to see what metonymical patterns these words instantiate, and what entities they refer to. I will discuss both of these stages in a bit more detail.

1.2.1 Metonymy recognition

Typically, metonymy recognition proceeds in one of three ways. Most approaches in computational semantics identify a word as metonymical when it violates selectional restrictions (see e.g. Pustejovsky, 1995; Copestake and Briscoe, 1995). In a sentence such as

(1.15) The ham sandwich is waiting for his check (Markert and Nissim, 2002a, p.1), the verb *wait* selects for an animate entity as its subject. Since *the ham sandwich* does not fulfil this requirement, it undergoes a metonymical shift of meaning to *the customer that ordered the ham sandwich*. This view of metonymy, however, fails to address a whole range of metonymies that do not violate selectional restrictions (see e.g. Markert and Hahn, 2001). In the earlier example (1.1), for instance, *Nixon* would not be recognized as metonymical, since Nixon was a human being capable of bombing Hanoi. A more robust theory of metonymy recognition is thus required.

Such a computational theory was developed by, among others, Markert and Hahn (2001). They "reject the assumption that metonymic language stands for a deviation from language norms and instead propose a mechanism which computes literal and metonymic interpretations *independently* from SRVs [selectional restriction violations]" (Markert and Hahn, p.146). Indeed, this framework tries to spot metonymies by taking the broader discourse context into account, which often contains direct or indirect clues about the metonymical character of a word. In (1.16), for instance, it is the current focus on a hard disk that helps point towards the metonymical use of *die Quantum*. This focus also guides its interpretation:

(1.16) "In der Leistung konnte die LPS 105 ebenfalls weitestgehend überzeugen. Laut Core-Test2.8 erreicht *die Quantum* eine mittlere Zugriffszeit von 16.5 ms, [...]"
("The performance of the LPS 105 [known to be a hard disk developed by Quantum, K.M. & U.H.] was mostly convincing. According to Core-Test2.8, *the Quantum* achieves an average access time of 16.5 ms, [...].")
(Markert and Hahn, p.149)

Hence, Markert and Hahn argue for a framework that takes into account the interdependencies between anaphora and metonymy resolution.

A third possible way of recognizing metonymies is the use of corpus-based learning techniques. This strategy, which was adopted by Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005), sees a possibly metonymical word as a polysemous target word that is ambiguous between a literal and a number of pre-defined metonymical meanings. It then appeals to machine learning algorithms to disambiguate the word. This study is discussed in more detail in section 1.3.

1.2.2 Metonymy interpretation

Once a word is recognized as metonymical, it still has to be interpreted. Markert and Hahn (2001) tackle this problem by constructing a search algorithm that relies on an extensive knowledge base. Their knowledge base is restricted to the IT domain, and contains relationships between the several entities in this domain, such as computers and printers. For each metonymy, the algorithm traces all these relationships and recovers all possible interpretations of the word. It then consults discourse constraints (see example 1.16) in order to pin down the most appropriate interpretation.

However, because of this algorithm's limitation to the IT domain, every extension to the system requires the addition of a new knowledge base. This constitutes an unrealistic enterprise for any large-scale metonymy resolution system. It is thus desirable to find an algorithm that can do without such world knowledge.

One possible solution is offered by Utiyama et al. (2000), who developed an algorithm that relies on corpus statistics instead. They note that in Japanese, contiguity relations between two entities *A* and *B* are often expressed by the phrase *A no B* (equivalent to English *B of A*), or by the presence of *A* and *B* within the same sentence (*A near B*). Statistics related to these two syntactic relations (*Q*s) can therefore be used to determine possible targets (*B*s) for a metonymy of the form 'Noun *A* Case-Marker *R* Predicate *V*'. The appropriateness of each target *B* is thus defined by:

$$\begin{aligned}
&(1.17) & L_Q(B|A,R,V) &= P(B|A,Q,R,V) \\
&(1.18) & &= \frac{P(A,Q,B,R,V)}{P(A,Q,R,V)} \\
&(1.19) & &= \frac{P(A,Q,B)P(R,V|A,Q,B)}{P(A,Q)P(R,V|A,Q)} \\
&(1.20) & &\approx \frac{P(B|A,Q)P(R,V|B)}{P(R,V)}
\end{aligned}$$

The last step of this decomposition assumes that (A,Q) and (R,V) are independent of each other. Because its denominator is constant, the appropriateness of each target can be determined by the calculation of two probabilities only.

The first of these calculations simply consists of counting the frequency of (A,Q,B) and dividing it by the frequency of (A,B):

$$P(B|A,Q) = \frac{f(A,Q,B)}{f(A,Q)} = \frac{f(A,Q,B)}{\sum_B f(A,Q,B)} \tag{1.21}$$

The second calculation requires two new equations: one is applied when the frequency of (B,R,V) is bigger than 0, the other when it is 0 (see equation 1.22). While the first of these is trivial, the second deserves some comment. This is because the non-occurrence of (B,R,V) in the training corpus should not lead to a zero probability: after all, the target B and predicate V are much more loosely related than the target B and the vehicle A. Therefore, if the combination (B,V) is absent in the corpus, equation (1.22) replaces B by the semantic classes to which it belongs, and weights each class by the probability that B belongs to it:

$$P(R,V) = \begin{cases} \frac{f(B,R,V)}{f(B)} & \text{if } f(B,R,V) > 0 \\ \frac{\sum_{C \in Classes(B)} P(B|C)\, f(C,R,V)}{f(B)} & \text{otherwise} \end{cases} \tag{1.22}$$

Utiyama et al. (2000) tested the robustness of their approach on a set of seventy-five metonymies from the literature on cognitive linguistics, psycholinguistics and computational linguistics. They used a 153 million word corpus taken from the Mainichi Newspaper as their training corpus, and based their semantic classes on a Japanese thesaurus. They found that fifty-three of the seventy-five metonymies were interpreted correctly. This performance was reached by taking both *no* and *near* relations into account. The *no* relation by itself led to fifty correct interpretations; the *near* relation to forty-three.

Although these results are quite promising, there are still a few problems. First of all, these figures are based on metonymies taken from the linguistic literature. Such examples are often clearer and more easily interpretable than those typical of everyday language. Second, the approach may not easily transfer to another language. Utiyama et al. (2000) themselves note that Japanese *no* roughly corresponds to English *of*, but that it can refer to many more (metonymical) relations. It therefore has to be determined whether English has equally informative constructions, in particular because co-occurrence relations have proven not to be very robust.

In short, there have already been a small number of computational approaches to metonymy recognition and interpretation. The most promising of these rely on corpus-based techniques to find or to interpret a metonymical word. With respect to metonymy recognition, these techniques were applied successfully by Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005), to whose study I now turn.

1.3 Metonymy resolution as a classification task

Markert and Nissim's study (see Markert and Nissim, 2002a, 2005b; Nissim and Markert, 2003, 2005) started from the crucial insight that metonymy resolution can be seen as a classification task (Markert and Nissim, 2002a). From this perspective, any possibly metonymical word such as a country or an organization name belongs either to the literal class, or to one of a number of pre-defined metonymical patterns. The resolution task is therefore related to Word Sense Disambiguation, and can be approached from a machine learning perspective.

1.3.1 Word Sense Disambiguation

Word Sense Disambiguation is usually defined as "the task of assigning sense labels to occurrences of an ambiguous word" (Schütze, 1998, p.97). The similarity to metonymy resolution is obvious: in our case, the ambiguous word is a possibly metonymical word such as *Nixon*, and the sense labels are `literal` and all pre-defined metonymical patterns. There are, however, two differences between metonymy resolution and classic WSD. First of all, theoretically speaking, the set of possible readings of a metonymical word is open-ended (see e.g. Nunberg, 1978). Yet, as Lakoff and Johnson

(1980) and many other linguists have argued, in practice, metonymies stick to a small number of metonymical patterns. The typical readings of a possibly metonymical word can thus be determined in advance. Second, classic WSD algorithms take training instances of one particular word as their input and then disambiguate test instances of the same word. Metonymy resolution algorithms, in contrast, "can take a set of labelled training instances of *different words belonging to one semantic class* as input and assign literal readings and possible metonymic patterns to new test instances of *possibly different words of the same semantic class*" (Markert and Nissim, 2002a, p.2). This is a major advantage: it makes the ambiguity problem less severe and removes the need for an annotated training set of every possibly metonymical word.

Markert and Nissim's (2002a) insight about the similarity between metonymy resolution and WSD opens up a multitude of possible automatic approaches to the resolution task. Just like classic WSD, these approaches can be subdivided into four types: supervised, dictionary-based, unsupervised and semi-supervised algorithms, as described in Jurafsky and Martin (2000).

The most robust approaches to Word Sense Disambiguation are supervised (see e.g. Leacock et al., 1998; Yarowsky, 2000). This means they need to be trained on a large corpus in which the ambiguous words of interest carry a semantic label. During this training phase, the system learns as much as possible about the association between the available features and the meaning of the target word. It should then be able to generalize from these training examples, and apply its newly acquired knowledge to the classification of unseen test examples. Well-known supervised algorithms include Naive Bayes classifiers, decision lists and nearest neighbour methods.

However, the robustness of supervised approaches does not come for free: semantic annotation is a labour-intensive and time-consuming process. For large-scale WSD systems, manual annotation is simply infeasible. Therefore other methods have been developed which try to minimize or avoid this annotation step. One possible strategy is the use of a machine-readable dictionary or thesaurus. This approach was pioneered by Lesk (1986), who simply measured the overlap between the context of an ambiguous word and all of its sense definitions in a dictionary. The definition with the highest overlap was then selected as the correct one. This simple approach has later been improved upon (see e.g. Guthrie et al., 1991), and still represents one of the most successful algorithms to WSD.

Even though no manual annotation is involved, dictionary- or thesaurus-based approaches still require an external knowledge source. This requirement is dropped by so-called unsupervised approaches, which merely need unannotated data. During the training phase, they cluster the feature vectors of the ambiguous words, and they assume that each cluster represents one of the target word's senses. The classification of a test instance then proceeds by computing its distance from (or similarity to) each cluster, and assigning it to the nearest one. Note, however, that these approaches differ from classic WSD in that they merely discriminate between several senses instead of directly disambiguating the target word: they do not automatically tell us which cluster represents which sense. One popular unsupervised algorithm was developed by Schütze (1998), and is discussed at length in chapter 2.

Unfortunately, unsupervised approaches tend to be less robust than supervised ones (Gaustad, 2004). Therefore researchers have looked into algorithms that strike a happy medium between these two (see e.g. Hearst, 1991; Yarowsky, 1995). These algorithms are called semi-supervised and start off with just a handful of sense-tagged training examples. A classifier is trained on these examples, and is made to tag a large set of unlabelled training data. From this set, it picks the instances whose classification it is most certain of, and it adds them to the training set. The classifier is then retrained, again tags the unlabelled set, selects new training data, and so on. This iterative algorithm allows the development of a classifier whose performance increases step by step.

In short, there is a whole spectrum of WSD algorithms that differ in the kind of training data and knowledge sources they need. Supervised algorithms require most human effort, unsupervised algorithms and dictionary-based the least, and semi-supervised algorithms sit in between the two extremes. Thanks to Markert and Nissim's (2002a) insight that metonymy resolution can be seen as a classification task, we know that all these WSD techniques can be applied to metonymy resolution as well.

1.3.2 Markert and Nissim's (2005a) annotation scheme

Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005) take a supervised approach to metonymy resolution. They therefore constructed a number of annotated corpora with target words from two semantic classes: location names (see Markert and Nissim, 2002b) and organization names (see Nissim and Markert, 2005). They extracted their possibly metonymical words, each with three sentences of context, from the British National Corpus (BNC)[1]. The corpora that I will use in this dissertation consist of 1000 mixed country names, 1000 instances of the country name *Hungary*, and 1000 mixed organization names.

In section 1.1, I already introduced the main metonymical patterns in Markert and Nissim's (2005a) annotation scheme. In addition to these class-specific patterns, the scheme also contains general patterns such as `object-for-name`, which can apply to words in all classes. In example (1.23), for instance, *Hungary* does not stand for the country or its people, but for the name itself, of which some linguistic properties are given:

(1.23) We could then say that, for example, "*Hungary*" is phonemically while "hungry" is; it would then be necessary to say that the vowel phoneme in the phonemic representation is not pronounced as a vowel, but instead causes the following consonant to become syllabic.

Finally, there are a `mixed` and an `othermet` category. The former covers those cases where a word has two different readings, while the latter applies to unconventional metonymies that do not belong to any of the categories above. Example (1.24) represents a mixed case: Denmark, Ireland and Belgium are referred to as "countries" (`literal`), but at the same time they are said to "indicate that they remain opposed" (`place-for-people`). Example (1.25) is an unconventional metonymy: *Hungary* and *Czechoslovakia* do not refer to the countries of those names, but rather to these countries' economies or certain firms in these countries.

[1] The data from this study is publicly available and can be downloaded from http://homepages.inf.ed.ac.uk/mnissim/mascara. From now on, all examples will be taken from this data, unless otherwise indicated.

	countries		Hungary	
reading	N	%	N	%
`literal`	737	79.7	746	75.9
`place-for-people`	161	17.4	201	20.4
`place-for-event`	3	0.3	14	1.4
`place-for-product`	0	0.0	0	0.0
`object-for-name`	0	0.0	1	0.1
`mixed`	15	1.6	14	1.4
`othermet`	9	1.0	7	0.7
total	925	100.0	983	100.0

Table 1.1: The distribution of metonymies in Markert and Nissim's (2002b) location data.

(1.24) Three countries — *Denmark*, *Ireland* and *Belgium* — meanwhile indicated yesterday that they remain opposed to another key element, which forsees the abolition of all limits on tax-paid goods that can be carried across borders by private travellers .

(1.25) Investing heavily in eastern Germany and more cautiously in *Hungary* and *Czechoslovakia*, it expects another good showing from Germany in 1992 as housebuilding progresses despite economic slowdown.

Tables 1.1 and 1.2 show how these readings are distributed in the studied corpora. Literal readings obviously dominate, but the relative frequency of metonymies stresses the need for metonymy resolution systems in NLP. The low frequency of the category `othermet` further strengthens the case for viewing metonymy resolution as a classification task, since it proves that the number of metonymies that do not belong to the pre-defined patterns is extremely small.

Markert and Nissim (2002b) and Nissim and Markert (2005) performed several experiments in order to test the reliability of their semantic annotation. They carried out the annotation independently, and afterwards measured reliability with the kappa-statistic. This yielded a result of κ=87.0% for the country metonymies and κ=89.4% for the organization metonymies (Nissim and Markert). Both annotations can thus be considered

	organizations	
reading	N	%
literal	622	64.3
organization-for-members	188	19.4
organization-for-product	66	6.8
organization-for-facility	14	1.4
organization-for-index	6	0.6
organization-for-event	1	0.1
object-for-name	6	0.6
object-for-representation	1	0.1
mixed	50	5.2
othermet	13	1.3
total	967	100.0

Table 1.2: The distribution of metonymies in Nissim and Markert's (2005) organization data.

very reliable. A Gold Standard was compiled from the instances that both researchers agreed upon after discussion. It is this Gold Standard that I will use for all experiments in this dissertation. Apart from a semantic label, each possibly metonymical word in the corpora also received a number of grammatical tags. These give the head(s) and the syntactic role(s) of the target word, and will prove crucial to the success of metonymy recognition[2]. The next sections present the results that were reached with this annotation.

1.3.3 Country results

Let us have a look at Markert and Nissim's (2002a) results on the country data, for which they used a decision list classifier. As a first step, Markert and Nissim studied what features are useful for this classification. They discuss three feature types. Co-

[2] At this moment, the official release of the data contains the grammatical annotation of the country and the Hungary corpora only. I therefore carried out the grammatical annotation of the organizations myself. This involved tagging each example for role, head, determiner and number of words.

occurrence features look at the content words in a context window around the target word, collocational features take into account the words immediately before or after the target, and grammatical features include the grammatical role of the target and its head. For the evaluation of the systems, Markert and Nissim computed a number of measures. Accuracy gives the percentage of all instances that the system classified correctly, taking all target readings into account. Precision, recall and F-score generalize over the metonymical target readings. Precision tells us what percentage of the words that were recognized by the system as non-literal are indeed metonymical, recall gives the percentage of non-literal words that were found by the system, and the F-score is the harmonic mean between these two[3]. All results were obtained by 10-fold cross-validation.

It was found that co-occurrences were the least reliable features. At best, the system reached an accuracy of 80.6%, with a precision of 55.4% and recall of 24.9% on the metonymical words. Collocational features led to precisions of up to 67.7%, but their recall scores never exceeded 18.5%. The grammatical features `role` and `role-of-head` ranked highest. Even with just three roles (`subject`, `object` and `other`), accuracy reached 84.3%, while precision and recall were 75.0% and 33.9%, respectively.

It was obvious that this last system could still be improved, and Nissim and Markert (2003) made two enhancements to tackle the low recall score in particular. First, they took more grammatical roles into account: `subject`, `passive subject`, `direct object`, `genitive`, `premodifier` and `pp modifier`. Second, they developed a new algorithm for test cases whose head was not seen in the training data. In those cases, the algorithm would iteratively search the training data for heads that belong to the same semantic class.

These semantic classes were defined on the basis of Lin (1998)'s thesaurus. This thesaurus contains the words that are most similar to a target word on the basis of their dependency relations in a newswire corpus. Nissim and Markert's (2003) algorithm, which is called `relax I`, now starts by trying to classify a test instance such as *subj-*

[3] In the rest of this dissertation, precision, recall and F-score always apply to the metonymical class.

algorithm	Acc	P	R	F
`hmr`	.817	.745	.186	.298
`relax I`	.851	.802	.410	.542
`relax II`	.859	.813	.441	.572
`combination`	.870	.814	.510	.627
`baseline`	.797	n/a	.000	n/a

Table 1.3: The results of Nissim and Markert's (2003) algorithms on the country data.

Acc	:	Accuracy
P, R, F	:	precision, recall and F-score for the metonymical class

of-lose. Since this particular example was not present in the training data, the classifier does not know what to do with it. Therefore `relax I` consults Lin's thesaurus, and finds that the most similar word to *lose* is *win*. It now replaces *subj-of-lose* by *subj-of-win* and again applies the decision list. This consultation phase is repeated until a head word is found that was present in the training data, or until the similarity score between the two words does not exceed a certain threshold anymore. Table 1.3 shows that this algorithm is indeed fairly successful: it increases precision with the head-modifier (`hmr`) feature from 74.5% to 80.2% and recall from 18.6% to 41.0%. Results on the Hungary data were similar: an F-measure of 62% was reached, more than 20% higher than the original 38.7% (Markert and Nissim, 2005b). The reasonable score of the `hmr` system in this case indicates that the restriction to one target word can indeed help performance. The advanced algorithms, however, smooth out this difference.

Nevertheless, the incompleteness of the thesaurus and the small number of training data meant the approach could still be improved. Therefore Nissim and Markert (2003) tested another algorithm, `relax II`, that backed off to the grammatical role if the precise constellation `role-of-head` was not found in the training data. As table 1.3 shows, this algorithm achieved a precision of 81.3% and a recall of 44.1%. The best performance, finally, was reached by a combination of `relax I` and `II`. This algorithm used `relax II` for subjects, and `relax I` for all other cases. It led to 81.4% precision, 51.0% recall and 62.7% F-score (see table 1.3) .

Feature	Description	Values
f1	grammatical role of target	subj, obj, ...
f2	lemmatised head/modifier of target	announce, shiny, ...
f3	determiner of target	def, indef, bare, demonst, other
f4	grammatical number of target	sing, plural
f5	# grammatical roles of target	1, more than 1
f6	# words in target	1, 2, 3, ...

Table 1.4: Nissim and Markert's (2005) features for the organization names.

1.3.4 Organization results

The classification of organization names (Nissim and Markert, 2005) proceeded in a similar fashion, but there are some minor differences. First, it used a slightly different feature set (presented in table 1.4). Nissim and Markert relied on their experience with the country data, so co-occurrence and collocational features were not tested. Second, examples that had more than one grammatical role were now represented by several feature vectors, one for each role. Third, `mixed` instances were removed from the training set, and test examples were only classified as `mixed` when their several feature vectors yielded different readings. Fourth, instead of decision lists, a Naive Bayes classifier was used. Finally, precision, recall and F-scores were computed for all of the classes involved.

The best classifier was the one that used all of the features in table 1.4. It reached an accuracy of 76.0%, thus beating the baseline by 11.7%, as shown in table 1.5. Performance on the most frequent metonymical patterns, members and product metonymies, was particularly promising, with F-scores of 68.1% and 58.0%, respectively.

The results on the country and organization data demonstrate that Markert and Nissim's study (see Markert and Nissim, 2002a, 2005b; Nissim and Markert, 2003, 2005) is a promising approach to metonymy resolution. At the same time, however, it does have some disadvantages. The most important of these is the manual labelling process that its supervised algorithms require, both for the semantic labels of the target words and for their syntactic relations. This labelling thwarts the generalization of

		literal			members			product		
	Acc	P	R	F	P	R	F	P	R	F
baseline	.643	.643	1.00	.783	n/a	0	n/a	n/a	0	n/a
NB	.760	.794	.903	.845	.670	.691	.681	.853	.439	.580

Table 1.5: Nissim and Markert's (2005) results for the organization names.

NB	:	Naive Bayes classifier
Acc	:	Accuracy
P, R, F	:	precision, recall and F-score for each of the classes

these particular classifiers to more metonymical patterns, and ultimately, to a wide-coverage metonymy resolution system. Moreover, Markert and Nissim's classifiers are still more concerned with metonymy recognition than interpretation. A classification such as `place-for-people` does not yet offer a full interpretation. After all, it does not tell us *what* people the metonymy refers to. Possible interpretations such as the population or the government of a country are situated on a lower level of Markert and Nissim's (2005a) annotation scheme. If the algorithms were tested on this level, they would certainly perform less well. Therefore a complete metonymy resolution system should consist of a recognition classifier like those in this section, complemented by an interpretation algorithm such as Utiyama et al. (2000)'s.

1.4 Discussion

This chapter introduced metonymy and some current approaches to metonymy resolution. I argued that most approaches to metonymy recognition suffer from a extensive need for human effort, either for the construction of knowledge bases (as in Markert and Hahn, 2001), or for the manual annotation of data (as in Markert and Nissim, 2002a, 2005b; Nissim and Markert, 2003, 2005). This knowledge acquisition bottleneck makes an extension of these classifiers to more metonymical patterns extremely problematic, because each of these would require the construction of new data sets or knowledge bases.

This dissertation is therefore mainly concerned with the development of an approach to metonymy recognition that reduces the current need for manual annotation. As I showed in this chapter, Markert and Nissim (2002a) argued that the task of metonymy recognition is comparable to that of Word Sense Disambiguation. This similarity paves the way for the possible application of a whole spectrum of machine learning approaches, most of which are still unexplored. In the next chapters, I will investigate three such learning algorithms, and study how they can make large-scale metonymy recognition more feasible.

Chapter 2

An unsupervised approach

Chapter 1 showed that metonymy recognition can be approached from a machine learning perspective. So far, however, the literature has tended to focus on labour-intensive supervised algorithms. This chapter therefore explores whether metonymy recognition can be tackled by unsupervised approaches. Section 2.1 introduces the particular algorithm I will use, Schütze's (1998) Word Sense Discrimination, and section 2.2 discusses its implementation in the SenseClusters program. Next, sections 2.3 and 2.4 present my first run of experiments on Markert and Nissim's (2002b) and Nissim and Markert's (2005) data, and their results. Finally, sections 2.5 and 2.6 will try to increase the classifier's initial performance by adding more training data and grammatical tags.

2.1 Schütze's (1998) Word Sense Discrimination

A popular approach to unsupervised WSD is Schütze's (1998) Word Sense Discrimination. As its name implies, this approach is able to discriminate automatically between the several senses with which an ambiguous word can be used. It is inspired by Miller and Charles' (1991) observation that humans rely on contextual similarity in order to determine semantic similarity. On this basis, Schütze (1998) hypothesized that there must be a correlation between contextual similarity and word meaning as well: "a sense is a group of contextually similar occurrences of a word" (Schütze, p.99). His article turns this intuition into an automatic algorithm.

The first step of this algorithm maps all words in the training corpus onto *word vectors*, which contain frequency information about their first-order co-occurrents. This simply means these vectors tell us how often each co-occurrent was present in one of the contexts of the word. The second step of the algorithm then zeroes in on the ambiguous target words. Since contextual similarity is the key to the algorithm, this step builds a vector representation of each of the contexts of the target. This is done by adding up the word vectors of the words that appear within a specified context window of 25 words around the target. Hence, these context vectors do not directly encode what words appear in the context of the target (first-order co-occurrence), but rather, what words appear in the context of the target's co-occurrents (second-order co-occurrence). The dimensionality of the vector space is subsequently reduced with SVD (Golub and Van Loan, 1989). Next, the context vectors are grouped into a pre-defined number of clusters. Each of these clusters is assumed to represent one of the senses of the target, according to the hypothesis above. The centroids of these clusters are therefore called *sense vectors*.

The classification of a test word is now trivial. The algorithm first determines what words occur in its context and sums together their word vectors to give the target's context vector. It then computes the similarity (the cosine) between this context vector and each of the sense vectors it discovered in the training data. Finally, it selects the most similar sense vector and assigns the test instance to the corresponding cluster.

The technical details of this algorithm obviously allow for extensive variation. In his paper, Schütze (1998) examined the results of a few such variations. First, he compared two types of feature selection: a local and a global one. The latter simply uses the most frequent words in the corpus as dimensions of the vector space, while the former only selects those words that appear in the context of the ambiguous target word. Schütze found that for pseudowords, global feature selection is more successful than its local counterpart, probably because of data sparseness in the latter case. For the polysemous words in table 2.1, the two types of feature selection gave a more similar performance.

Second, Schütze (1998) also argued it is a good idea to reduce the number of dimen-

word	Acc.	baseline	word	Acc.	baseline
capital	94%	64%	space	76%	56%
interest	93%	58%	suit	95%	57%
motion	87%	55%	tank	92%	90%
plant	70%	54%	train	74%	74%
ruling	91%	60%	vessel	98%	69%

Table 2.1: The results of Schütze’s (1998) Word Sense Discrimination.

sions in the vector space with a technique such as Singular Value Decomposition. This technique abstracts away from the word dimensions, and is claimed to discover underlying semantic features instead. Therefore the vector space no longer suffers from lexical problems such as synonymy (where two dimensions incorporate the same concept) or polysemy (where one dimension represents several concepts), and the algorithm should be able to compute contextual similarity more reliably.

Third, Schütze (1998) compared the relative merits of statistical and frequency-based feature selection. Frequency-based selection takes into account the *n* most frequent words in the corpus (global) or in the context of the ambiguous word (local). Statistical selection, in contrast, uses a χ^2-test to select those words whose presence is correlated with the presence of the ambiguous word. This assumes that “candidate words whose occurrence depends on whether the ambiguous word occurs will be indicative of one of the senses of the ambiguous word and hence useful for disambiguation” (Schütze, p.102). It was found that statistical selection does better with SVD, but that it is outperformed by frequency-based selection when no SVD is used.

A final issue, which was not examined by Schütze (1998), is the clustering algorithm. Schütze used Buckshot (Cutting et al., 1992), a combination of the EM algorithm and agglomerative clustering. The problem with EM is its sensitivity to local optima, and therefore another clustering algorithm may well perform better.

The results of Schütze’s (1998) approach are promising. Table 2.1 shows the accuracy of the best system for each of the ten ambiguous words on which the algorithm was tested. With about 8,000 training instances on average, this accuracy clearly beats the baseline in nine out of ten cases. Even those baselines that exceed 60%, as in Markert and Nissim's (2002b) and Nissim and Markert's (2005) data, do not present the algorithm with major difficulties.

Yet, some critical reflections are necessary. First, Schütze's (1998) algorithm was evaluated on just ten naturally polysemous words. This is an absolute minimum, and it implies that the results may not generalize to new data sets. Schütze himself noted that "in the future more extensive test sets will be required to establish the general applicability of disambiguation algorithms" (Schütze, p.117).

Second, Schütze's (1998) algorithm makes use of topical similarity in its discrimination between word senses. Text topics, however, are not very informative where possible metonymies are concerned. Irrespective of the topic covered by a text — tourism or politics, say — the country name *Hungary* can be used either literally or metonymically. This may indicate that Schütze's basic algorithm will not be robust enough to deal with Markert and Nissim's (2002b) and Nissim and Markert's (2005) data. Nevertheless, the combination of its co-occurrence information with syntactic relations, which were found to be useful by Markert and Nissim (2002a), may provide a possible solution. After all, Schütze already anticipated that it might be necessary to "incorporate other, more structural constraints [...] to achieve adequate performance for a wide variety of ambiguous words" (Schütze, p.117). It is precisely these two challenges, the extension of the algorithm to a new class of ambiguous words and the incorporation of syntactic information, that this chapter will address.

2.2 SenseClusters

Schütze's (1998) approach is implemented in the SenseClusters package (see e.g. Purandare and Pedersen, 2004a,b; Kulkarni and Pedersen, 2005), which I will use for the experimental part of this chapter below. This package also incorporates some interesting variations and extensions to the algorithm. The most significant of these concern the nature of the selected features on the one hand, and the clustering algorithm on the other.

As we have seen, Schütze (1998) compiled context vectors by summing together the word vectors of the co-occurrents within a context window of 25 words on either side of the target. This idea is taken further by Purandare and Pedersen (2004a,b) and Kulkarni and Pedersen (2005), which introduce a number of new ways to compile second-order context vectors: *bigrams*, *co-occurrences*, and *target co-occurrences*. Bigrams are "ordered pairs of words that co-occur within five positions of each other" (Purandare and Pedersen, 2004b, p.2), co-occurrences are unordered bigrams, and target co-occurrences are "co-occurrences that include the given target word" (Purandare and Pedersen, p.2). While it is still the word vectors of individual words that get summed together, their dimensions depend on the actual feature that is used.

Purandare and Pedersen (2004b) showed that bigrams give the biggest gain in performance over Schütze's (1998) original algorithm. Since I will use this bigram algorithm throughout the rest of the chapter, it is worth exploring it step by step. The algorithm starts with the compilation of a bigram matrix. The rows of this matrix "represent the [bigram's] first word and the columns represent the second word" (Kulkarni and Pedersen, 2005, p.2). The cells give either frequency information, saying how often the words corresponding to their row and column occur together in the training data, or statistical information, indicating how closely the presence of the words is correlated. The word vectors with this information are then added up in order to give the second-order context vectors, which finally get clustered.

This clustering stage represents the second of SenseClusters' extensions. Purandare and Pedersen (2004b) found that a hybrid algorithm called Repeated Bisections performs better than Schütze's (1998) algorithm, at least for sparse data. Repeated Bisections combines a hierarchical clustering approach with a partitional one. It starts off with all instances in one cluster (hierarchical), but iteratively splits this cluster on the basis of the partitional K-means algorithm. Again, I will replace Schütze's basic algorithm with this extension.

The final stage in the SenseClusters pipeline — evaluation — deserves some comment as well. One of the problems with unsupervised approaches is that they may be able to identify several sense clusters in the data, but that they cannot tell us which cluster represents which sense. Evaluation therefore proceeds indirectly: SenseClusters automatically finds the alignment of senses and clusters that leads to the fewest misclassifications — this is the confusion matrix that maximizes the diagonal sum.

In order to test its extensions to Schütze's algorithm, SenseClusters has been evaluated on the SENSEVAL-2 data, and on the *line*, *hard* and *serve* corpora. Purandare and Pedersen (2004b) constructed a train and test set for all words in these corpora with at least 90 training instances. This left 24 SENSEVAL-2 words — each with between 90 and about 250 training instances — as well as all three words in the *line*, *hard* and *serve* corpora — with 1615, 2356 and 2356 training instances respectively. As I described above, evaluation proceeded indirectly, through the optimal alignment of clusters and labels. Of all algorithms tested, the combination of both extensions (bigrams and repeated bisections) reached the highest F-scores on the SENSEVAL-2 data, but fell short on the bigger *line*, *hard* and *serve* corpora.

In spite of these general patterns, the results on the particular words are very diverse. The bigram algorithm scores particularly well for words with a low baseline, attaining F-scores of 55.34% on *art* (baseline 46.32%), of 64.76% on *facility* (baseline 48.28%), and of 53.47% on *leave* (baseline 38.18%). Whenever the baseline lies above 50%, however, the bigram algorithm fails to beat it. This occurs with words such as *child* (baseline 56.45%, F-score 55.17%), *live* (baseline 57.63%, F-score 41.82%) and *blind* (baseline 82.46%, F-score 79.17%). Apart from one exception, the other algorithms, which rely on co-occurrence instead of bigram information and different clustering algorithms, do not outperform the baseline either. This is an indication that the algorithm in its present form may not be robust enough to deal with Markert and Nissim's (2002b) and Nissim and Markert's (2005) data, whose baselines lie well above 60%. The next section investigates if this is indeed the case.

2.3 Experiments with the raw data

In a first round of experiments I tested the algorithm above on Markert and Nissim's (2002b) raw location data, without adding any additional syntactic information. 60%

of the instances were used as training data, 40% as test data. I limited myself to the Hungary data, because the second-order context vectors rely solely on co-occurrence information. Since different country or organization names tend to co-occur with different words, they are less suitable for a co-occurrence-based classifier. In all experiments I set the number of pre-defined clusters to two. I did this because, apart from the two main readings `literal` and `place-for-people`, all other senses are represented by just a handful of instances. It would be unreasonable to expect the algorithm to identify clusters corresponding to these senses. Indeed, if the algorithm is run with more clusters, it returns a solution that is very heterogeneous with respect to the sense labels. I therefore worked with two clusters, while keeping the infrequent metonymies in the training and test sets, so that performance measures could be compared more easily with Markert and Nissim's (2005b) figures.

My initial experiment used the precise algorithm described in Purandare and Pedersen (2004b). It takes a context window of 20 words on either side of the target, selects bigrams with a log-likelihood score of 3.841 or more, compiles the context vectors, applies SVD to reduce the number of dimensions to 300, and clusters the resulting vectors using Repeated Bisections. After this initial experiment, I varied the algorithm on three dimensions: the size of the context window, the use of SVD and the statistical test. This was done with three specific research questions in mind:

- **Are smaller context windows better than large ones?**
  Markert and Nissim (2002a) discovered that, with co-occurrence features, the reduction of window sizes from 10 to about 3 led to a radical improvement in precision (from 25% to above 50%) and recall (from 4% to above 20%). I will test if the same phenomenon occurs with unsupervised learning.

- **Does Singular Value Decomposition result in better performance?**
  Schütze (1998) found that his algorithm clearly performs better with SVD than without. However, there are reasons for investigating if this is also the case with metonymies. SVD is said to abstract away from the word dimensions, and to discover topical dimensions instead, but as Markert and Nissim (2002a) argue, the sense distinctions between the literal and metonymical meanings of a word

are not of a topical nature. Therefore I will repeat the experiments without SVD.

- **Should the features be selected by a statistical test?**
  Purandare and Pedersen (2004b) used a log-likelihood test to select their features, but Schütze (1998) claims that features can best be chosen on the basis of their frequency instead. I will therefore perform the experiments with and without the log-likelihood test.

2.4 Results on the raw data

Table 2.2 presents the results of the experiments described above. Two patterns immediately catch the eye. First of all, the accuracy never beats the majority baseline of 77.35%[1]. Second, there seems to be a trade-off between the general accuracy and the F-score for the metonymical class: the F-score tends to go down as accuracy goes up, and vice versa.

Let us evaluate these results in the light of the experimental questions. For a start, the influence of context size is difficult to determine. In the default system, accuracy increases with smaller contexts, but this goes hand in hand with a decreasing F-score. The same goes for the the system without statistical test (-LL, +SVD). For the (-LL, -SVD) system, smaller contexts bring down accuracy, while the overall effect on the F-scores is less clear. The (+LL, -SVD) results do not display a clear pattern either. Generally speaking, the highest accuracy is reached with small contexts, but contexts of a more intermediate size lead to higher F-scores. This is probably the case because smaller context windows do not allow the system to discover any bigrams that typically co-occur with metonymies, which I assume are less frequent than those co-occurring with literal target words. Therefore the algorithm tends to throw most of the data into one big cluster, and recognizes only a small number of metonymies, if any. The result is a relatively high accuracy, but a low F-score.

The second question concerned the effect of SVD. From a comparison between the two

[1] Note that only accuracy can be compared to this baseline. A baseline system that classifies all instances as literal returns a recall of 0% for the metonymical class, and no precision or F-score.

	+LL, +SVD		+LL, -SVD		-LL, +SVD		-LL, -SVD	
	Acc	F	Acc	F	Acc	F	Acc	F
baseline	77.35	n/a	77.35	n/a	77.35	n/a	77.35	n/a
20	64.88	14.92	67.43	21.25	54.96	33.08	62.34	18.29
15	66.92	11.03	60.31	14.61	57.51	35.43*	49.87	39.87**
12	68.19	8.76	48.60	38.10**	57.76	30.52	62.34	31.63
10	49.62	26.62	49.87	33.33	59.03	28.18	62.85	25.77
7	72.26	n/a	51.65	33.81	72.26	n/a	54.71	30.52
5	66.92	7.14	52.16	35.71	72.26	n/a	57.25	29.31
3	72.26	n/a	60.05	23.76	72.26	n/a	55.73	27.35

Table 2.2: The results of four algorithms with varying context sizes on the raw Hungary data.

+LL	:	statistical feature selection
-LL	:	frequency-based feature selection
+SVD	:	dimensionality reduction with SVD
-SVD	:	no dimensionality reduction
**	:	result is significantly better than random assignment of data to clusters
*	:	difference between result and random assignment approaches significance

first systems, it is clear that dropping SVD has a positive effect on F-scores, which go up with all context sizes. Again, in five of the seven systems this is accompanied by a decrease in accuracy. I suspect that the higher F-scores can be attributed to the fact that the algorithm now works with word dimensions instead of “topical” dimensions. As I have noted before, the co-occurrents of a possibly metonymical target word are better indications of its meaning than the largely irrelevant topical distinctions discovered by SVD. However, skipping SVD makes the algorithm work in far more dimensions, and leads to extreme data sparseness. This may explain the lower accuracy of these systems.

With larger contexts, the effect of SVD is comparable to that of the statistical test: removing this step from the algorithm improves F-scores, but brings down accuracy.

This may be due to the relative infrequency of metonymical data and the resulting small number of statistically significant features. In the case of smaller contexts, however, skipping SVD leads to better results than skipping the statistical test. I assume this is because the number of features within these small windows is so low that the literal features outnumber the metonymical ones when a frequency-based criterion is used. Finally, dropping both SVD and the statistical test leads to the same effect as dropping either of them: accuracy goes down, F-scores go up.

As I mentioned above, the algorithm's accuracy seems to be negatively correlated with its F-score for the metonymical class. Whenever one goes up, the other goes down, and vice versa. The explanation is straightforward: a high accuracy typically results from one large cluster, which covers most of the training and test instances. Just a small number of instances is assigned to the smaller second cluster, and typically hardly any of these are metonymical. In contrast, when the algorithm identifies two clusters of roughly the same size, many more instances will end up in this second cluster. The literal instances that do so bring down accuracy, while at the same time the metonymical instances here boost the F-score.

Therefore we have to ask ourselves to what degree this F-score is just an accidental result from the size of the two clusters. In other words, is this F-score higher than it would be in a system that divided the instances among its two clusters randomly? This question can be answered by a χ^2-test. By comparing the experimental results to the expected (random) results, this statistical test checks if there is a correlation between two variables, in our case the clusters and the sense labels. If it finds that the two are independent, the experimental result is no better than a random assignment of instances to its two clusters would be. If instead the test finds that the two variables are correlated, the result is either better or worse. For our F-score to significantly beat this random baseline, the χ^2-test must return a significance level smaller than 0.05, and the number of metonymies in the smaller cluster must be higher than expected.

With $\alpha = 0.05$, these two requirements are fulfilled by only two of the results above[2]. These are the (+LL, -SVD) result with a context size of 12 ($Acc = 48.60$, $F = 38.10$,

[2] For all statistical tests in this dissertation, $\alpha = 0.05$.

	cluster 1	cluster 2
LIT	138	166
MET	29	60

Table 2.3: Confusion matrix of the +LL, -SVD algorithm with a context size of 12

	cluster 1	cluster 2
LIT	140	164
MET	26	63

Table 2.4: Confusion matrix of the -LL, -SVD algorithm with a context size of 15

$\chi^2 = 4.623$, $df = 1$, $p = 0.032$) and the (-LL, -SVD) result with a context size of 15 ($Acc = 49.87$, $F = 39.87$, $\chi^2 = 8.001$, $df = 1$, $p = 0.005$). In addition, one other result approaches significance ($p < 0.10$, see table 2.2). The confusion matrices of the two most successful systems are shown as tables 2.3 and 2.4. It is immediately clear that the proportion of metonymical instances in the second cluster is much bigger than that in the first cluster. Remarkably, of all the experiments above, these are the systems with the lowest accuracy. This shows that accuracy does not necessarily tell us something about a system's ability to recognize metonymies. In spite of their low accuracy, these two systems are the only ones that successfully identify a literal and a metonymical cluster.

These experiments have thus shown that the unsupervised approach tested here is not yet robust enough to produce reliable metonymy recognition. This does not mean, however, that we should dismiss it altogether: there are a number of strategies that might improve the algorithm's initial performance. One involves adding extra data, another tags the available data with grammatical information. I now turn to these possible solutions.

	+LL, +SVD		+LL, -SVD		-LL, +SVD		-LL, -SVD	
	Acc	F	Acc	F	Acc	F	Acc	F
baseline	77.35	n/a	77.35	n/a	77.35	n/a	77.35	n/a
15	61.83	30.48	55.98	34.88	58.78	31.30	55.22	32.54
12	61.83	29.81	55.98	32.26	60.81	29.25	57.51	31.28
10	65.90	27.78	66.16	26.59	60.56	26.34	55.98	29.29
7	63.10	28.43	58.97	30.97	50.38	35.71	57.95	32.77
5	64.12	29.74	60.57	30.99	51.65	32.06	54.57	31.58
3	64.89	31.09*	59.36	34.65	52.42	33.21	57.43	38.05**

Table 2.5: The results on the Hungary data of four algorithms with a training set of 13 million words and varying context sizes.

+LL : statistical feature selection
-LL : frequency-based feature selection
+SVD : dimensionality reduction with SVD
-SVD : no dimensionality reduction
** : result is significantly better than random assignment of data to clusters
* : difference between result and random assignment approaches significance

2.5 Experiments with an extended training set

Adding extra data has often proved to be a successful way of improving the performance of NLP systems. Our Hungary sets in particular are extremely small, and may not provide enough information for the compilation of robust word vectors. Therefore I extended the training set with data from the BNC, making sure not to add any of the occurrences of *Hungary* in the test set. This resulted in a large data file that contained 874 instances of Hungary, and about 13 million words. I subsequently repeated the experiments above — their results can be seen in table 2.5.

It is clear from this table that even the large training set I used is not able to improve on the earlier results. Admittedly, system performance now is more stable. Accuracy generally lies between 55% and 65%, with F-scores somewhere between 25% and 35%.

However, only once does the algorithm output two clusters whose correlation with the target meanings is statistically significant; one other time this correlation approaches significance. We can therefore conclude that adding extra training data does not lead to a higher level of robustness.

There are a number of reasons why this may be the case. First there is the noise in the training data. Remember that Markert and Nissim's (2002b) data were carefully controlled for noise: homonyms or instances that could not be classified were removed from the training set. This is not the case for the large training set: the BNC data was added without any human intervention, and may therefore contain much more noise than Markert and Nissim's data sets. Second, random corpus data may be less helpful to the algorithm than words that appear close to the target word. Finally, it may simply be the case that co-occurrence information is not sufficient for reliable metonymy recognition, irrespective of the number of training instances that is taken into account. If this is correct, extending the training data with grammatical information may succeed where simple co-occurrences fail.

2.6 Experiments with grammatical information

The addition of grammatical information is a second possible way of improving system performance. Schütze (1998) already noted that the disambiguation of some types of polysemy may require structural information in addition to co-occurrence statistics, and Markert and Nissim (2002a) observed that dependency information was absolutely necessary for reliable metonymy recognition. This section will therefore investigate what happens if grammatical tags are added to the words in the training file.

The addition of syntactic information to unsupervised systems has been studied in particular with relation to Latent Semantic Analysis (LSA). Its merit is an object of ongoing debate. The original proponents of vector-based techniques often claim that syntactic information should be disregarded (see e.g. Landauer et al., 1997; Lund et al., 1995), and there are indeed indications that LSA performs less well when part-of-speech tags are added to the words (see e.g. Wiemer-Hastings and Zipitria, 2001).

	+LL, +SVD		+LL, -SVD		-LL, +SVD		-LL, -SVD	
	Acc	F	Acc	F	Acc	F	Acc	F
baseline	77.35	n/a	77.35	n/a	77.35	n/a	77.35	n/a
20	75.32	2.02	67.94	8.89	75.83	2.06	74.05	3.81
15	75.06	2.00	52.04	25.00	75.06	2.00	74.49	9.17
12	74.81	1.98	53.69	32.82	75.06	2.00	74.23	7.41
10	74.55	1.96	57.00	33.20	74.81	1.98	74.74	3.92
7	74.05	3.81	61.13	32.26*	74.05	5.66	74.55	3.92
5	59.80	30.56	49.94	34.01	72.26	13.01	74.23	3.88
3	61.83	35.02**	58.44	27.49	67.94	22.64	69.43	20.69

Table 2.6: The results of four algorithms with varying context sizes on data with grammatical tags only.

+LL : statistical feature selection
-LL : frequency-based feature selection
+SVD : dimensionality reduction with SVD
-SVD : no dimensionality reduction
** : result is significantly better than random assignment of data to clusters
* : difference between result and random assignment approaches significance

However, Wiemer-Hastings and Zipitria found that performance goes up if the original bag-of-words model is replaced by a structured representation of sentences that reflects dependency information. Sahlgren (2002) even formulates a “plea for linguistics”, in which he argues that “we need to think hard about how to incorporate more linguistic information into the vector representations” and that we must “move beyond the bag-of-words” approach (Sahlgren, p.5).

In order to test the effects of adding grammatical information, I parsed Markert and Nissim’s (2002b) data with the RASP parser (Briscoe and Carroll, 2002). I represented the output of this dependency parser in two ways. The first of these replaced context words with their grammatical roles (subject, object, modifier, etc.). The second added this dependency role as a tag to each word. Any word for which RASP did not

	+LL, +SVD		+LL, -SVD		-LL, +SVD		-LL, -SVD	
	Acc	F	Acc	F	Acc	F	Acc	F
baseline	77.35	n/a	77.35	n/a	77.35	n/a	77.35	n/a
20	48.60	30.32	63.87	27.84	75.06	2.00	55.73	31.15
15	55.73	29.79	59.80	31.86*	75.06	2.00	56.74	32.37
12	55.47	35.52*	63.87	32.20*	74.81	5.77	54.45	31.50
10	57.00	33.20	66.67	31.91**	74.30	7.41	54.20	29.72
7	57.25	32.91	60.56	38.37**	71.50	12.70	59.54	27.91
5	58.02	33.76	57.07	20.29	69.21	22.37	62.40	37.72**
3	63.61	35.68**	52.71	32.43	63.63	35.55**	67.88	36.36**

Table 2.7: The results of four algorithms with varying context sizes on data with words and grammatical tags.

+LL : statistical feature selection

-LL : frequency-based feature selection

+SVD : dimensionality reduction with SVD

-SVD : no dimensionality reduction

** : result is significantly better than random assignment of data to clusters

* : difference between result and random assignment approaches significance

find a grammatical role was left unreplaced or untagged. I subsequently repeated all experiments from section 2.3.

Table 2.6 indicates that the first context representation was not very successful: only one system gave a significant correlation between clusters and meanings. Large contexts proved to be particularly useless for metonymy recognition. This is because most of the dependency relations that RASP returns are modifier relations. As a result, these dominate in large contexts of literal as well as metonymical words, so that SenseClusters is not able to distinguish between them. The best algorithms are therefore those that use smaller contexts and a statistical test, which helps them find informative features (and not modifiers, for instance).

The second context representation presents a handy solution to the modifier problem: if the individual words are tagged, SenseClusters will look at the entire `word/tag` entity, and not just at the tags. The predominance of modifier relations should thus be less problematic. Table 2.7 shows that this is indeed the case. It is immediately clear that these experiments are much more successful than the ones above. No fewer than six F-scores significantly beat the random baseline and another three tend towards significance. It is moreover striking that the successful algorithms all have accuracies above 60% — the accuracies of the best algorithms in section 2.3 did not exceed 50%. Clearly, classifiers that rely on grammatical information are able to combine good F-scores with a reasonable accuracy.

Without a doubt the most successful algorithms are those that do not use SVD (the second group). Four of the nine systems whose χ^2-result reached or tended towards significance belong to this group. Although they have lower accuracies than those in the third group, they succeed in identifying two clusters that correlate with the two senses of the target word. Their F-scores are similar to those in the first and fourth groups, but their accuracies are higher in five out of seven cases.

It has become clear that the addition of grammatical information helps unsupervised algorithms distinguish between the two sense clusters of possibly metonymical words. This finding is compatible with Markert and Nissim's (2002a) claim that dependency information is crucial to metonymy recognition. Moreover, vector-based unsupervised algorithms work best with word dimensions, demonstrating that it is the words themselves, and not underlying dimensions, that give the most valuable information about the meaning of a possibly metonymical target word.

2.7 Discussion

Figure 2.1 compares the best results from this chapter (according to the χ^2-test) with Markert and Nissim's (2005b) results, and highlights the general outcomes of the experiments. First, the systems with grammatical tags added to the words predominate in the top ten, occupying no fewer than six places. Of the systems that use this informa-

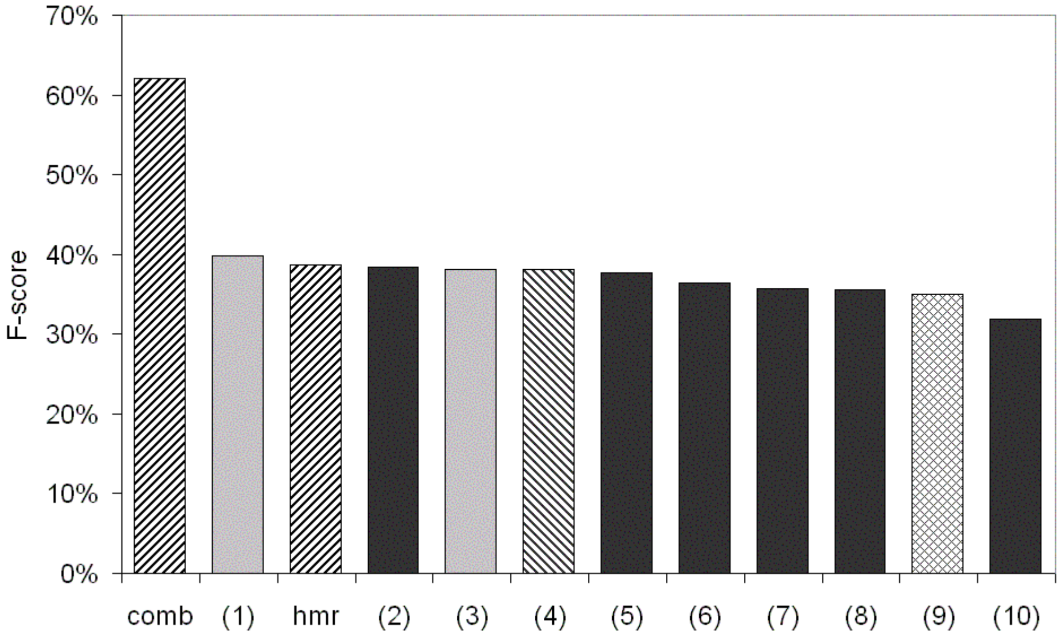

Figure 2.1: A comparison of the best unsupervised results with Markert and Nissim's (2005b) algorithms.

comb	:	Markert and Nissim's (2005b) combination algorithm			
hmr	:	Markert and Nissim's (2005b) head-modifier algorithm			
(1)	:	raw,	-LL,	-SVD,	context 15
(2)	:	word/gramm,	+LL,	-SVD,	context 7
(3)	:	raw,	+LL,	-SVD,	context 12
(4)	:	13m,	-LL,	-SVD,	context 3
(5)	:	word/gramm,	-LL,	-SVD,	context 5
(6)	:	word/gramm,	-LL,	-SVD,	context 3
(7)	:	word/gramm,	+LL,	+SVD,	context 3
(8)	:	word/gramm,	-LL,	+SVD,	context 3
(9)	:	gramm,	+LL,	+SVD,	context 3
(10)	:	word/gramm,	+LL,	-SVD,	context 10

tion, the (+LL, -SVD) algorithms with intermediate context sizes and the (-LL, -SVD) algorithms with small context sizes have the highest chance of success. Other good results were reached by the best systems on the raw data, which take up two places in the top three. Second, seven of the best algorithms worked in word dimensions, which again stresses that SVD tends to form an impediment to metonymy recognition.

How do these systems compare with previous results? Unfortunately, Markert and Nissim (2002a) did not publish any co-occurrence results for the Hungary data, as they did with the mixed country names. On this latter set, their classifier reached an F-score of 34.36% when it relied on co-occurrence information only. This lies below our performance, but the difference must be attributed at least partially to the fact that data with mixed country names necessarily contain a wider variety of co-occurrences.

Some results for the Hungary data were given by Markert and Nissim (2005b), however. With the head-modifier feature, they obtained an F-score of 38.7%. This figure, which was reached with 10-fold cross-validation, is slightly below my best unsupervised algorithm. Markert and Nissim's combination algorithm, which incorporates extra semantic information and grammatical back-off, led to an F-score of 62% and lies head and shoulders above the unsupervised results. All accuracies in Markert and Nissim moreover consistently beat the baseline, which proved impossible for the unsupervised algorithms in this chapter. In short, although the F-scores of the unsupervised algorithms compare favourably to Markert and Nissim's most basic results, overall these approaches are considerably less robust.

Unsupervised algorithms undoubtedly provide less reliable metonymy recognition than their supervised competitors. They moreover have the disadvantage of needing one classifier for each possibly metonymical word. Yet, they can prove very beneficial as a pre-processing step for extracting data that need to be annotated. Such a first step, which automatically discriminates between a literal and a metonymical cluster, can reduce annotation effort. Instead of labelling every possibly metonymical word, the annotators now simply have to go through the initial classification and correct errors. As a reduction of human participation in metonymy recognition, unsupervised algorithms thus certainly have their usefulness. The most promising systems are those that drop SVD and take grammatical information into account.

Chapter 3

A Supervised Approach

Chapter 2 showed that unsupervised algorithms can be made to distinguish between a metonymical and a literal cluster of senses when the training and test data carry grammatical tags. However, the performance of these unsupervised systems lies far below the results in Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005). The next step in our comparison of different learning algorithms therefore involves the application of a supervised classifier that should be able to replicate Markert and Nissim's and Nissim and Markert's results. The classifier I will use is an implementation of memory-based learning. This approach is supervised, like the classifiers in the literature, but its learning stage is much more simple. Section 3.1 discusses the theory behind memory-based learning and its implementation in TiMBL. Section 3.2 presents my first experiments, which use the same features as Markert and Nissim and Nissim and Markert. Section 3.3 explores the results of adding semantic classes to the data, while section 3.4 tries to replace the manual grammatical labels with automatic ones.

3.1 Memory-based Learning

The central hypothesis underlying memory-based learning (MBL) says that

> "performance in cognitive tasks is based on reasoning on the basis of similarity of new situations to *stored representations of earlier experiences*, rather than on the application of *mental rules* abstracted from earlier experiences" (Daelemans et al., 2004, p.19)

Because of this, memory-based learning has also been dubbed "lazy learning". In contrast to the decision list and Naive Bayes classifiers of Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005), an MBL classifier eschews the formulation of complex rules or the computation of probabilities during its training phase. Like *k*-nearest neighbour (*k*-NN) systems, it classifies a test instance by comparing it to the most similar training instances.

This classification procedure consists of two components: a learning component and a performance component, as described in Daelemans et al. (2004). During the learning component, the MBL classifier simply stores all training examples in its memory, in the form of a feature vector and a label. In the performance component, the system tries to classify an unseen feature vector. It computes the distance between this vector and all training vectors and simply returns the most frequent label of the most similar training examples.

Let us have a closer look at this performance component. First we have to note how the system computes the distance between two vectors, and second, how it resolves ties. I will succinctly describe how these two steps are implemented in TiMBL, short for Tilburg Memory Based Learner (Daelemans et al., 2004), the classifier that I used in my experiments.

TiMBL's IB1 algorithm computes the distance or similarity between two vectors X and Y by adding up the weighted distances δ between their corresponding feature values:

$$\Delta(X,Y) = \sum_{i=1}^{n} w_i \delta(x_i, y_i) \tag{3.1}$$

The weights for each feature are determined by equation (3.2), which divides the feature's Information Gain by its split info, the entropy of its feature values:

$$w_i = \frac{H(C) - \sum_{v \in V_i} P(v) \times H(C|v)}{si(i)} \tag{3.2}$$

$$si(i) = -\sum_{v \in V_i} P(v) log_2 P(v) \tag{3.3}$$

The numerator in equation (3.2) is the Information Gain of feature *i*. It measures "how much information it [this feature] contributes to our knowledge of our class label" (Daelemans et al., 2004, p.20). Its first term, $H(C)$, is the entropy or uncertainty of all class labels. Its second term is the uncertainty of these class labels given the feature values of feature i with values V_i, where each feature value is weighted by its probability. The numerator gives the difference between these two uncertainties, and hence, the amount of information that lies in feature *i*. The problem with Information Gain, however, is that it "tends to overestimate the relevance of features with large numbers of values" (Daelemans et al., p.21). These features are very informative about the training examples, but do not generalize well to new test instances. Therefore equation (3.2) divides the Information Gain by the entropy of the feature values (equation 3.3), which increases with the number of features.

When the *k* nearest neighbours have been determined, it is possible that a tie between several class labels occurs. TiMBL breaks this tie by incrementing *k* by one, and adding the extra nearest neighbours to the voting set. If the tie still persists, TiMBL backs off to the most frequent class label in the training data.

In short, the ideas underlying memory-based learning are rather intuitive and simple, and the system is therefore much "lazier" than the classifiers in Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005). The next sections investigate whether it is able to perform equally well.

3.2 First experiments

In this first stage of supervised experiments I performed some tests that are similar to those in Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005). In order to find out if memory-based learning is able to replicate their results, I evaluated all algorithms with 10-fold cross-validation.

3.2.1 Head-modifier features

In chapter 1 I discussed the features that were found to be informative by Markert and Nissim (2002a), and I mentioned that the classifiers benefited from grammatical features in particular. The simplest algorithm in Nissim and Markert (2003) combined two instances of grammatical information — the role of the target word and its head — in one head-modifier (`hmr`) feature, `role-of-head`. Obviously, the performance of this system was rather modest. On the country data it achieved an accuracy of 81.7% and an F-score of 29.8%, while the F-score on the Hungary data was 38.7% (see table 3.1[1]). This classifier served as the inspiration for my first TiMBL experiments.

The results of my experiments, which are given in table 3.1, are almost identical to Nissim and Markert's (2003) and Markert and Nissim's (2005b) results. The country figures mirror them perfectly, while my F-score on the Hungary data lies only slightly higher (40.13% vs. 38.7%). These results come as no surprise — the use of a single feature prevents any difference between the algorithms from showing up.

The present figures, and the F-scores in particular, are rather low. This is because the only metonymies that are recognized as such are those whose head-modifier relation was already present in the training data. If it was not, the set of nearest neighbours will contain all training data, and the instance will be classified as literal. As a result, the system recognizes very few metonymies (hence the low recall), but if it returns a metonymical reading, it mostly does so correctly (hence the high precision). Note, finally, that the accuracy of all three systems already beats the baseline. Moreover, the

[1] This table contains all available results from Nissim and Markert (2003) and Markert and Nissim (2005b) for this algorithm. Other results have not been published.

		baseline	Acc	P_{met}	R_{met}	F_{met}
countries	N&M	79.68%	81.7%	74.5%	18.6%	29.8%
	TiMBL		81.73%	74.47%	18.62%	29.78%
Hungary	M&N	75.89%	—	—	—	38.7%
	TiMBL		80.67%	81.82%	26.58%	40.13%
organizations	TiMBL	65.44%	69.58%	86.29%	26.82%	40.92%

Table 3.1: Results with the `hmr` feature from TiMBL, Nissim and Markert (2003) and Markert and Nissim (2005b).

baseline: accuracy of the majority baseline

Acc: accuracy of the tested algorithms

P_{met}, R_{met}, F_{met}: precision, recall and F-score for the metonymical class

N&M: Nissim and Markert (2003)

M&N: Markert and Nissim (2005b)

Hungary classifier compares favourably to the unsupervised algorithms in the previous chapter: none of these reached an F-score of 40%.

3.2.2 Backing off to grammatical roles

The problem with the previous system is that it considers all training data whenever an exact match of the test instance is absent. We can avoid this by introducing a second feature, which contains only the grammatical role of the word. If the `hmr` feature value is not present in the training data, the set of nearest neighbours will now consist of all training instances with the same grammatical role as the test instance. Therefore the classifier will back off to the majority reading of this grammatical role. This should have the clearest effect on recall scores, as the system will assign metonymical labels more often. This approach corresponds to Nissim and Markert's (2003) algorithm `relax II`.

In addition to these `role` and `hmr` features, the Hungary and country words received a third feature that indicates if the target word has a second head, and a fourth feature

		baseline	Acc	P_{met}	R_{met}	F_{met}
countries	N&M	79.68%	85.9%	81.3%	44.1%	57.2%
	TiMBL		86.38%	81.48%	46.81%	59.46%
Hungary	TiMBL	75.89%	84.54%	80.13%	51.05%	62.37%
organizations	TiMBL	65.44%	75.64%	80.43%	55.64%	65.78%

Table 3.2: Results with grammatical back-off from TiMBL and Nissim and Markert (2003).

baseline: accuracy of the majority baseline

Acc: accuracy of the tested algorithms

P_{met}, R_{met}, F_{met}: precision, recall and F-score for the metonymical class

N&M: Nissim and Markert (2003)

that contains this head. This is because the presence of two heads, and the identity of the second head, may contain information about the class of the word. This brings the total number of features for the Hungary and country data to four.

In Nissim and Markert (2005), each organization name received four extra features: its number of grammatical roles, its grammatical number, its number of words, and the nature of its determiner (if present). Nissim and Markert showed their classifier worked best with all of these features, and consistently performed less well if one of them was left out. However, the number of words in the organization name led to lower results in my experiments, so I only added three features.

Table 3.2 shows the results of this second round of experiments. The introduction of new features clearly leads to a better performance for all data sets. As predicted, it is the recall scores in particular that benefit from the higher number of features. For all three classifiers, a single-sided t-test shows that the improvement in recall is indeed statistically significant. This is also the case for all three accuracies. The effect on precision is less clear-cut: it increases by 7% for the country metonymies, but goes down by 1.7% for the Hungary data and by 6% for the organization names — none of these differences are statistically significant. Again, my results closely correspond to those reached by the `relax II` algorithm in Nissim and Markert (2003) and Markert and

		baseline	Acc	P_{met}	R_{met}	F_{met}
countries	N&M	79.68%	87.0%	81.4%	51.0%	62.7%
	TiMBL		86.59%	80.17%	49.47%	61.18%
Hungary	M&N	75.89%	—	—	—	62%
	TiMBL		84.74%	80.39%	51.90%	63.08%

Table 3.3: The best results for the location data from TiMBL, Nissim and Markert (2003) and Markert and Nissim (2005b).

baseline: accuracy of the majority baseline

Acc: accuracy of the tested algorithms

P_{met}, R_{met}, F_{met}: precision, recall and F-score for the metonymical class

N&M: Nissim and Markert (2003)

M&N: Markert and Nissim (2005b)

Nissim (2005b). My recall score on the countries is slightly higher, but the difference is not statistically significant.

3.2.3 Final improvements

The second round of experiments already exhausts all information in the annotation files. Still, a third round of experiments indicates that it may be possible to use this information in a more beneficial way. This time the changes to the previous approach are minimal. I reached the best results on the Hungary and country metonymies after replacing the `role-of-head` by a `role` feature. The role and the head of a target word are thus seen as independent of each other. As a result, recall and F-scores are higher for both data sets, although the differences are not statistically significant. As table 3.3 shows, the best results in Nissim and Markert (2003) and Markert and Nissim (2005b) are very similar to mine. For the country data, my F-score lies about 1.5% lower, while that of the Hungary data lies 1% higher. This is particularly striking since Nissim and Markert's and Markert and Nissim's best results were obtained by a combination of the `relax I` and `relax II` algorithms, and thus incorporate semantic information. The TiMBL results, in contrast, were achieved without any semantic information at all.

In the last organization experiment, I followed Nissim and Markert (2005) in their treatment of `mixed` instances. These instances are not very informative for the classifier. Mostly a subgroup or all of their feature values are clearly indicative of a class such as `members`, while their label is `mixed`. Therefore Nissim and Markert removed these confusing examples from the training data. The `mixed` instances in the test data were kept, but if a target had two or more grammatical functions, its features were presented to the classifier in several feature vectors, each corresponding to one grammatical role. If the classifier assigned two vectors of the same example to different classes, the classification was treated as `mixed`. I thus processed the data in the same way, and found the results in table 3.4.

While TiMBL's performance on the `literal` and `member` classes is similar to that in Nissim and Markert (2005), that on `product` and `mixed` metonymies is very different. Both discrepancies may well be due to the small number of instances in these categories and my own labelling of the data, however. They could be caused by small differences in the treatment of determiners or plural words (for the `product` metonymies), and that of instances that have several grammatical roles (for the `mixed` cases), for instance. These results are therefore less important than those for the bigger categories.

The treatment of the `mixed` category in Markert and Nissim's (2005b) approach deserves some extra comment, however. Its problem is that the feature vectors often point towards one reading only, so that we cannot expect the classifier to recognize these vectors as `mixed`. Consider the following example:

(3.4) Sun could move the manufacture of these parts from TI, which began it quite successfully, to *Fujitsu*, which has been very anxious for the business.

In this sentence, the literal reading of *Fujitsu* modifies the preposition *to*, while its `organization-for-members` reading is the subject of *has been*. However, because of the limited annotation scheme, only the former grammatical role makes it into the feature vector. The classifier therefore returns the label `literal`, which is the correct classification as far as the feature vector is concerned. More syntactic information should thus be taken up in the annotation scheme to resolve the present mismatch between this scheme and the semantic categories.

	Acc	P_{met}	R_{met}	F_{met}
N&M	76.0%	—	—	—
TiMBL	76.12%	80.43%	56.78%	66.57%
		P	R	F
literal	N&M	79.4%	90.3%	84.5%
	TiMBL	78.71%	92.04%	84.86%
members	N&M	67.0%	69.1%	68.1%
	TiMBL	66.51%	66.21%	66.36%
product	N&M	85.3%	43.9%	58.0%
	TiMBL	91.94%	57.33%	70.49%
mixed	N&M	46.7%	28.0%	35.0%
	TiMBL	27.27%	5.26%	8.82%

Table 3.4: The best results for the organization data from TiMBL and Nissim and Markert (2005).

baseline: accuracy of the majority baseline

Acc: accuracy of the tested algorithms

P_{met}, R_{met}, F_{met}: precision, recall and F-score for the metonymical class

N&M: Nissim and Markert (2005)

3.2.4 Error analysis

In order to improve the results above, we first have to understand what mistakes the classifier makes. These mistakes can be subdivided into three broad categories: they can be caused by a lack of syntactic information, a lack of semantic information, or a lack of world knowledge.

The first important category of mistakes is due to missing syntactic information. These mistakes occur throughout all grammatical categories. Take the prepositional phrases as a first example. In all three data sets, the majority of prepositional phrases have a literal reading. There are exceptions, however, as the following examples show:

(3.5) My Government will further encourage the development of democratic institutions and market economies in central and eastern Europe; and pursue the completion of Association Agreements with *Hungary*, Poland and Czechoslovakia.

(3.6) It was then examined for commercial potential by both ICI (who eventually marketed it under the name Terylene) and *DuPont* (who called it Dacron).

The country and organization names in these examples are `place-for-people` and `organization-for-member` metonymies, but the algorithm is not able to recognize them as such. This is because the feature vectors contain only the grammatical role `pp` and the particular preposition. TiMBL therefore finds a large number of exact matches in the training data, but most of these represent literal readings. For example (3.5), 18 of the exact matches are literal, 16 are place-for-people, 1 is mixed and 1 is place-for-event. This pre-dominance of literal targets in prepositional phrases causes TiMBL to make tens of mistakes.

Yet, in most of these cases, the ambiguity can be resolved by attachment information. The PP in (3.5) modifies *Agreement*, and that in (3.6) is attached to *examined*. Both these words represent actions that can only be performed by people, and not by countries or organizations, and thus indicate that both examples are metonymical. Even though attachment information may not disambiguate every single instance, it should be helpful in most.

The lack of syntactic information also lies at the basis of some `othergramm` misclassifications. In Markert and Nissim's (2005b) approach, the label `othergramm` is given to all grammatical roles that cannot be termed subject, passive subject, object, indirect object, prepositional phrase, genitive or premodifier. Most of the instances involved have a literal reading (like example 3.7), but again there are exceptions to this rule (like example 3.8):

(3.7) By the sixth century they had begun to force their way into Gaul (France and *Belgium*), and there they eventually settled .

(3.8) Everybody is dancing on the grave of Drexel Burnham Lambert — everybody but entrepreneurial *America*, that is.

Because the literal readings predominate in the class `othergramm`, example (3.8) will be misclassified. Yet again, this misclassification can be prevented if (3.7) is labelled as `pp into`, and (3.8) as `subj dance`.

An extended annotation scheme could also be helpful for some less frequent errors. One example, that of the `mixed` readings, was already discussed above. Another is found with metonymies that function as the subject of a copula such as *to be*, like examples (3.9) and (3.10):

(3.9) Nigeria and *Ghana* were to be participants.

(3.10) *Hungary* is hopeful but stays sceptical of better relations.

Again, most subjects of *to be* are literal, and this is the reading to which both of these examples are wrongly assigned. This mistake could be avoided by introducing a feature for the predicate of the copula[2]. Both *participants* and *hopeful* are words that can only apply to people, indicating that the subjects of the copula must be metonymical.

The second important category of mistakes cannot be solved by adding syntactic information. Addressing these mistakes, which occur when a certain head feature was not seen in the training data, requires semantic knowledge instead. Consider examples (3.11) and (3.12):

(3.11) Under pressure from the European Community, the two countries have agreed to operate a temporary water management scheme, which aims to reconcile the Slovakian need for increased energy with *Hungary*'s fears about environmental impact.

(3.12) As *Ireland* opened up to foreign investment under de Valera's successor, Sean Lemass, another element in the value structure came to prominence [...].

In these examples, the target's heads (*fear* and *open up*) were not present in the training data. Therefore their nearest neighbours contain all genitives for (3.11) and all subjects for (3.12). However, the majority classes among these nearest neighbours do not corre-

[2] Because of data sparseness, the feature value should be a semantic class rather than the exact predicate.

spond to the class of the test instance: (3.11) is misclassified as `literal`, while (3.12) is wrongly recognized as `place-for-people`.

This time the solution to the problem does not lie in extra syntactic features, but in semantic information. If *fear* in (3.11) is labelled as *human feeling*, say, TiMBL will first look for nearest neighbours with the same semantic class. In this group the `people` reading of *Hungary* is likely to predominate. Similarly, it should be clear from the class of *open up* that this verb does not require a human agent. Such an approach, which is related to Nissim and Markert's (2003) `relax I` algorithm, will be investigated in the next section.

Finally, there are some mistakes which neither syntactic nor semantic information can solve. In example (3.13), it is world knowledge that tells us that the talks are about events concerning Hong Kong, and not about the territory itself or the people there. It goes without saying that solving these mistakes lies beyond our limited means.

(3.13) In his last assignment as Minister of State at the United Kingdom Foreign and Commonwealth Office, Francis Maude visited China on July 25-27, primarily for talks on *Hong Kong*.

Despite these mistakes, this section has shown that a "lazy" algorithm such as memory-based learning is able to replicate the results obtained by the more complex algorithms in Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005). Without relying on semantic information, the best TiMBL figures even approached the results of Nissim and Markert's (2003) `combination` algorithm, which does take semantic classes into account. An error analysis showed that the addition of such semantic information should make TiMBL perform even better. This is the topic of the next section.

3.3 Semantic information

Nissim and Markert's (2003) `relax I` algorithm incorporated semantic information by iteratively running through Dekang Lin's classes of semantically similar words (Lin,

1998) (see chapter 1). In this section I will test an approach that is more compatible with the ideas behind memory-based learning. I will add semantic classes to the feature vectors as one or more extra features, so that TiMBL can search its memory for training data whose heads belong to the same class as that of the test instance. These extra features are based on WordNet's hierarchy of synsets.

WordNet is a lexical database that, among others, structures English verbs, nouns and adjectives into a hierarchy of so-called "synonym sets" or synsets (Fellbaum, 1998). Each word belongs to such a group of synonyms, and each synset "is related to its immediately more general and more specific synsets via direct hypernym and hyponym relations" (Jurafsky and Martin, 2000, p.605). *Fear*, for instance, belongs to the synset *fear, fearfulness, fright*, which has *emotion* as its most immediate, and *psychological feature* as its highest hypernym. This tree structure of synsets thus corresponds to a hierarchy of semantic classes that can be used to add semantic knowledge to a WSD system like ours.

This addition of semantic knowledge can proceed in many different ways. For my system I experimented with a few constellations of semantic features. The simplest of these just takes the highest hypernym synset of a particular head, and adds it as an extra feature. If a word has several WordNet senses, only the most frequent sense is taken into account. A more complex approach combines all the hypernym synsets to which a head word belongs. Because the maximum number of hypernyms is 11, I added 11 new features to the vectors. The last of these represented the highest hypernym, the second-to-last contained the second highest, and so on. If the word did not have 11 hypernyms, the remaining features would just take the word's synset as their value. The result of this approach is that TiMBL looks for heads that are as closely related to the test head as possible. If it does not find a word within the same synset, it looks within the first hypernym synset. If it does not find a training instance there, it climbs another synset, and so on. This is the approach I expected to perform best, because it is able to make more fine-grained semantic distinctions than the previous one.

As table 3.5 shows, the benefits of WordNet information are questionable. It consistently brings down metonymical precision (without reaching significance, however),

		baseline	Acc	P_{met}	R_{met}	F_{met}
countries	original	79.68%	86.59%	80.17%	49.47%	61.18%
	wn all		85.62%	72.46%	53.19%	61.35%
	wn all wsd		85.73%	71.74%	52.66%	60.74%
	wn highest		86.59%	77.34%	52.66%	62.66%
	wn highest wsd		86.49%	75.76%	53.19%	62.50%
Hungary	original	75.89%	84.74%	80.39%	51.90%	63.08%
	wn all		84.44%	78.05%	54.01%	63.84%
	wn all wsd		84.94%	79.88%	55.27%	65.34%
	wn highest		84.13%	77.56%	51.05%	61.58%
	wn highest wsd		84.84%	78.53%	54.01%	64.00%

Table 3.5: TIMBL's results with WordNet features.

baseline: accuracy of the majority baseline

Acc: accuracy of the tested algorithms

P_{met}, R_{met}, F_{met}: precision, recall and F-score for the metonymical class

wn all: all WordNet hypernyms included

wn highest: only the highest WordNet hypernym included

wsd: manual disambiguation of heads

and although recall scores are mostly slightly higher, the differences are again not statistically significant and their effect on F-scores is minimal. The system with all hypernyms (wn all) worked best on the Hungary data; the system with the highest hypernyms (wn highest) proved better on the country data.

There are several possible reasons why semantic information is less beneficial than expected. First, I already pointed out that TIMBL's results without semantic information are almost as good as Nissim and Markert's (2003) and Markert and Nissim's (2005b) results *with* semantic information. It is therefore possible that there is not much gain to be found in semantic classes anymore. Second, there is the limited number of data for each grammatical role, which implies that few semantic classes will have enough members to really make a difference. This is particularly problematic with verbs, whose WordNet hierarchy is not very well elaborated, so that many verbs just sit in their own restricted synset, without any hypernym relations. Third, sometimes WordNet classes are heterogeneous with respect to the target readings. This is the case with the synset *psychological feature*, for instance. Even though this class seems to point towards a metonymical reading, its hyponym *model* in example (3.14) triggers a literal reading of *Hungary*.

(3.14) Italy, he jokes, should be *Hungary*'s *model*.

Finally, my algorithm for finding WordNet classes may also have introduced some noise. When the head word was polysemous, I selected the first, most frequent, WordNet sense. Obviously, not all head words are used with their most frequent sense, which may have led to a few misclassifications. Take example (3.15), for instance. The first WordNet meaning of *threaten* is "pose a threat to; present a danger to", which has *exist*, *be* as its only hypernym. The correct WordNet meaning, however, is "to utter intentions of injury or punishment against". This sense has *communicate, intercommunicate* among its hypernyms, a semantic class that is indicative of an animate agent. The selection of the first meaning is likely to result in a classification as `literal`; the selection of the second in the correct reading as `place-for-people`.

(3.15) *China* has always *threatened* to use force if Taiwan declared independence.

I have tested this last explanation by manually selecting the correct WordNet sense for all ambiguous head words in the data. The resulting (wsd) figures are again presented in table 3.5. They display a marginal improvement on the Hungary data, where the best F-score now lies above 65%, but a small drop in performance on the country data. Indeed, although there is a substantial number of heads that do not have their most frequent meaning, their disambiguation only rarely leads to a different classification of the target word. The added labour of manual disambiguation is certainly not matched by a parallel gain in performance.

In short, adding semantic information in the form of WordNet semantic classes does not increase the performance of our classifier. This may be due to the limited number of data in each of these classes, the heterogeneity of WordNet classes with respect to the target readings, or to the simple fact that semantic classes do not introduce much new information into the basic feature vectors.

3.4 The effects of parsing

It has become clear that the algorithm's quality depends primarily on grammatical features. So far these features have been based on manual annotation. This annotation, however, is a labour-intensive process, so that the practical development of a large-scale metonymy recognition system certainly requires the computerization of this step. In this section I will investigate the effects on performance of such a development. Like Nissim and Markert (2003), I used RASP, a robust dependency parser that outputs, among others, the grammatical role of a word and its head (Briscoe and Carroll, 2002).

In spite of its robustness, RASP is unable to analyze a considerable number of training instances. Its most striking mistakes can be categorized into seven types. First, RASP simply disregards words between brackets. This is not dramatic, however, as in Markert and Nissim's (2005b) approach most of these words would receive the label `othergramm` anyway. The second mistake, RASP's inability to deal with appositions, is more serious, because it often results in the recognition of the wrong dependency relation. According to the parser, *Hungary* in example (3.16) is a modifier of *have*.

(3.16) We also have association agreements with three eastern European countries — Poland, Czechoslovakia and *Hungary*.

Third, RASP tends to break down on long coordinations. In example (3.17), for instance, *Hungary* is wrongly recognized as a modifier of *begin*. Next, the parser fails to deal with ellipsis: the omission of the verb in example (3.18) means no head-modifier relation is discovered for the target word. Fifth, some genitival *'s* forms are wrongly seen as short forms of *is*, and their heads classified as subjects of be, as in example (3.19):

(3.17) On March 6 Frans Andriessen, the EC Commissioner responsible for external relations and trade policy and relations with other European countries, began a series of visits to Poland, Czechoslovakia, *Hungary*, Bulgaria and

Romania.

(3.18) Germany has accepted more than 200,000; Austria and *Hungary* 50,000 each; and Sweden 45,000.

(3.19) Earlier, on Nov. 4, *Hungary*'s then State Secretary for Foreign Affairs, Laszlo Kovacs, had announced that Hungary was ready to give assurances to the US that it would not use developed technology for military purposes.

The final two types of mistake are mainly due to the form of the BNC data. First, the BNC does not end a newspaper heading in a punctuation mark, as in example (3.20). This again causes RASP to break down, analyzing *Hungary* as a modifier of the preposition *of*. Second, the BNC also includes the names of newspaper features, as in example (3.21). Unsurprisingly, RASP cannot deal with those, and analyzes *Hungary* as a modifier of *anti-right*.

(3.20) Hungarian suspension of rouble-backed licences *Hungary* on Sept. 1, 1989, revalued the forint against the rouble at R1.00=F27.50.

(3.21) EUROPE HUNGARY Anti-right marches.

These seven types of mistakes make up an overwhelming majority of RASP's errors. Many of those difficult constructions — newspaper headings or long coordinations, for example — show up frequently because we are dealing with country names. They may therefore be less of an issue when other metonymical patterns are involved. Nevertheless, they do have an important effect on my results.

In order to determine how major this effect is, I did some new experiments with the best country and Hungary classifiers from section 3.2. Table 3.6 shows the results of these experiments, and compares them to the original figures. Nissim and Markert's (2003) algorithm seems slightly more robust than mine, particularly with regards to precision. It is not clear why this is the case. Possibly they pre- or post-processed the RASP data differently, or their classifier may be able to handle noisy data better than TiMBL.

The figures make it clear that manual annotation is crucial in obtaining high perfor-

		baseline	Acc	P_{met}	R_{met}	F_{met}
countries	N&M	79.68%	83.0%	64.0%	38.8%	48.3%
	TiMBL		81.95%	59.83%	37.23%	45.90%
	original		86.59%	80.17%	50.27%	61.79%
hungary	TiMBL	75.89%	79.25%	64.00%	40.51%	49.62%
	original		84.74%	80.39%	51.90%	63.08%

Table 3.6: Results with automatic grammatical annotation from TiMBL and Nissim and Markert (2003).

baseline: accuracy of the majority baseline

Acc: accuracy of the tested algorithms

P_{met}, R_{met}, F_{met}: precision, recall and F-score for the metonymical class

N&M: Nissim and Markert (2003)

mance. Both classifiers still beat the baseline, but their accuracy has dropped by around 5%. The effects on the scores for the metonymical class are even more radical: precision drops by 15% to 20%, recall by 11% to 13%, and F-score by 13% to 16%. The computerization of grammatical annotation thus seriously challenges the robustness of the systems developed above, and moreover provides my suggestion for an extended annotation scheme (see section 3.2.4) with practical problems.

3.5 Discussion

Figure 3.1 summarizes the results of the most important algorithms that I tested in this chapter. I started off with a very basic algorithm, which used one head-modifier feature, and which reached a modest F-score of around 30% for the country data and 40% for the Hungary data. This basic Hungary result is comparable with the highest unsupervised F-score that I reached in the previous chapter, and immediately indicates the success of a supervised learning approach such as memory-based learning. Next, I added a new feature that contained the grammatical role of the target word and thus provided a form of grammatical back-off. Its extremely positive effect on performance

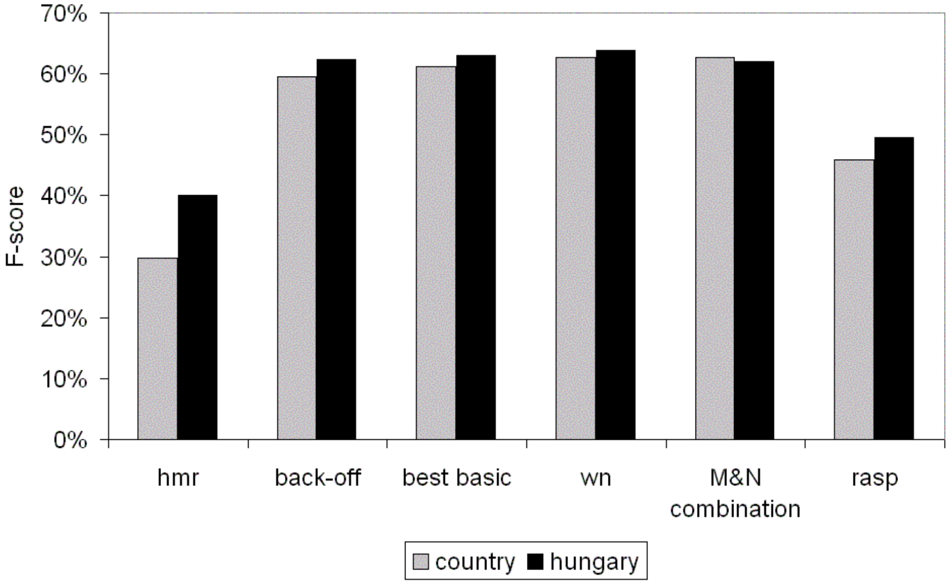

Figure 3.1: The F-scores of six supervised algorithms.

hmr	:	TiMBL with the head-modifier feature
back-off	:	TiMBL with grammatical back-off
best basic	:	TiMBL's best result without extra information
wn	:	TiMBL with WordNet information
M&N	:	Markert and Nissim's (2005b) combination algorithm
rasp	:	TiMBL with automatic grammatical tags

was clear in the resulting F-scores of around 60%.

So far, I had taken roughly the same steps as Nissim and Markert (2003), and my results had therefore been very similar to theirs. I then found that slightly higher scores could be reached by replacing the `role-of-head` by a `head` feature, or by adding semantic data from WordNet. Although performance increased for both the country and Hungary data, the differences never reached statistical significance, and adding semantic information was much less beneficial than expected. Nevertheless, as figure 3.1 reflects, my best basic algorithms reached a very similar performance to Markert and Nissim's (2005b) `combination` algorithm, but without relying on semantic information.

Finally, I questioned the robustness of supervised approaches to metonymy recognition by replacing the manual grammatical information with automatically obtained tags. This caused a performance drop of around 15%, indicating that at the present stage, supervised metonymy recognition still crucially relies on human annotation. The next chapter will therefore study if the amount of annotation can be reduced.

Chapter 4

A semi-supervised approach

The previous chapters have established that a supervised approach such as memory-based learning is better at tackling metonymy recognition than an unsupervised approach such as Schütze's (1998). However, in chapter 1, I already pointed out that the success of supervised approaches is compromised by their extensive need for manual annotation. This final chapter will therefore investigate whether this manual annotation can be reduced to an absolute minimum, by relying on a semi-supervised algorithm. Section 4.1 recaps the philosophy behind semi-supervised learning. Section 4.2 then investigates how TiMBL performs with a growing number of labelled training instances. Section 4.3, finally, compares these curves with the performance of a semi-supervised system that needs only a handful of labelled training examples.

4.1 Semi-supervised learning

Semi-supervised learning was already discussed briefly in chapter 1. It is a machine learning approach that tries to exploit a very small number of manually labelled training instances. These initial seeds are iteratively supplemented by training instances that are selected and tagged by the classifier itself.

Yarowsky (1995) summarizes the basic semi-supervised learning algorithm in five steps. The first step consists of identifying all instances of the target word in a corpus. Step two picks a number of so-called seed data, which are tagged manually, while the rest of the instances are kept as an untagged training pool. Step three consists of training the classifier on the seed set, and applying it to the training pool. The classifier then selects "those members in the residual that are tagged as SENSE-A or SENSE-B with probability above a certain threshold, and add[s] those examples to the growing seed sets" (Yarowsky, p.4). This step is repeated iteratively, until step four stops it. Step five, finally, tests the classifier on an unseen set of data. This iterative procedure allows the development of a classifier whose performance keeps improving while the seed set grows.

Yarowsky (1995) reported some very good results with this algorithm. With just two seed words he reached an average accuracy of 90.6% on twelve test words. I will now examine whether a similar approach can be applied to metonymy recognition.

4.2 Learning curves

Before investigating how a semi-supervised algorithm affects performance, we should have a look at the performance that can be reached with a subset of the labelled training data. The resulting learning curves will provide a type of Gold Standard against which we can evaluate the semi-supervised algorithm.

Like in chapter 2, I again split the data into a training set (60%) and a test set (40%). I developed a simple algorithm that iteratively added 10% of the total number of training instances to the current training set. On every iteration TiMBL was trained on this growing training set and tested on the held-out test set. I did this experiment for both the country and the Hungary data. The results can be seen in figures 4.1 and 4.2.

Both figures show that even a small portion of the training data can lead to a reasonably high performance. With the "raw" grammatical features from section 3.2.3, a random 10% of the data led to an F-score of 41% on the country metonymies, and 32% on

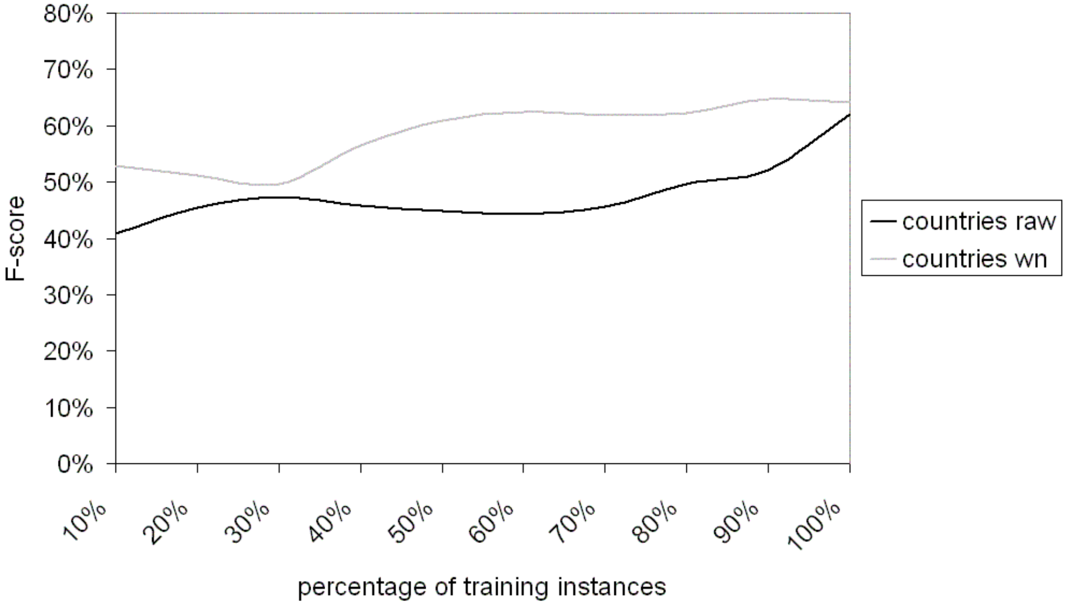

Figure 4.1: Learning curves for the country data with and without WordNet (wn) information.

the Hungary metonymies. This performance increased as more and more training instances were added, reaching a final F-score of 62% on the country, and 68% on the Hungary metonymies.

The figures moreover exhibit two striking features. When we add all WordNet (wn) hypernyms to the feature vectors, the learning curve for the Hungary data is comparable with that of the raw data. It starts lower, but makes up for this difference and ends at a very similar F-score. For the country data, however, the F-score with WordNet classes constantly lies above that without. With around 60% of the data, the difference even amounts to 17%. This indicates that, contrary to the findings in chapter 3, semantic information is extremely helpful with a smaller number of training instances. Even though it is now less likely that the head of a test instance was present in the training data, TiMBL is able to find training heads that belong to the same class. Indeed, 60% of the training data with this semantic information give the same F-score on the metonymical class as 100% of the data without. The absence of this phenomenon on the Hungary data can be explained by its smaller variety in heads: the country data

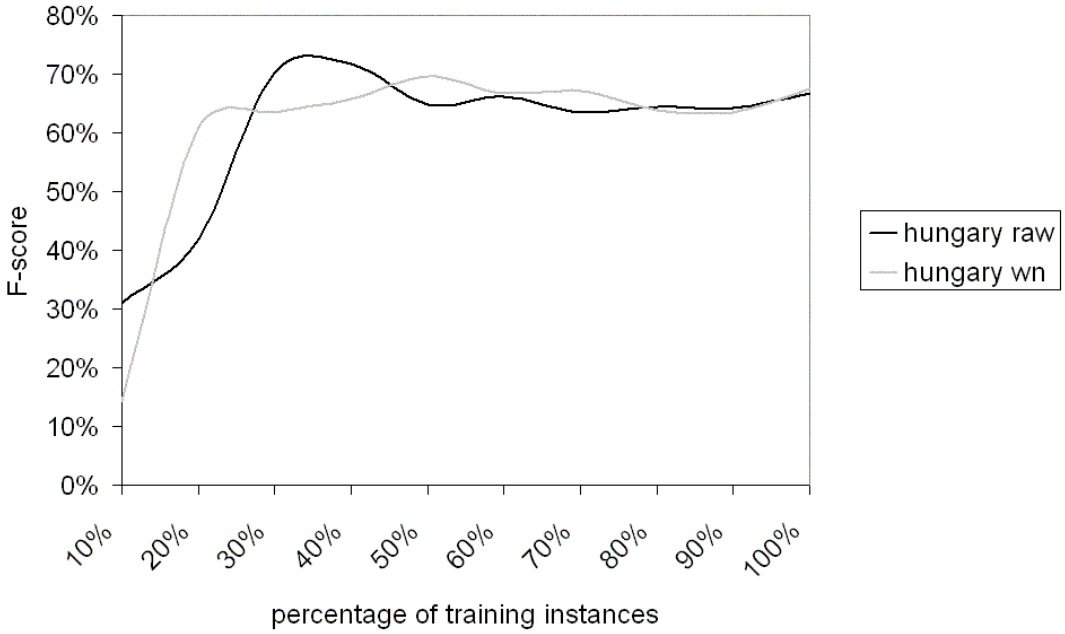

Figure 4.2: Learning curves for the Hungary data with and without WordNet (wn) information.

contains 317 different heads, while the Hungary data has only 216.

The learning curve for the Hungary data, however, displays another interesting feature. It shows that the maximum score on this data was reached with around 30% to 40% of the training instances. These F-scores of around 72% lie 4% higher than the final F-score, indicating that a small number (200 to 250) of training instances may lead to a higher performance than a larger training set.

In short, it is clear that even a small number of labelled training instances can lead to a very high performance. With a mixed category such as the country names, semantic information helps resolve the lack of identical semantic heads; with one target word such as *Hungary*, less than half of the data may contain more interesting information than the full training set. This paves the way for the development of a semi-supervised algorithm.

4.3 Semi-supervised experiments

As we saw in section 4.1, a semi-supervised training algorithm starts with a limited number of seed instances and an unlabelled training pool. My implementation randomly selected 5 literal and 5 metonymical instances from the training set as its seed data; the rest of the data was set aside as the training pool. Next, TiMBL was trained on this seed data and labelled the entire training pool.

The next step involved the selection of new training data, based on the classifier's confidence. For each classification, TiMBL's confidence can be measured by the distance between the instance and the closest seed. Instances with a smaller distance to one of the seeds are more likely to be classified correctly than those with a bigger distance. Therefore the algorithm reads through the TiMBL output, registers all distances, selects the smallest, and adds the corresponding feature vectors to the training set, together with their labels returned by TiMBL. Since many examples may have the same distance, the group that is added to the seed data can be rather large. This procedure is repeated until the entire training pool has been labelled and added to the training set. Because the initial seeds were chosen randomly, I performed each of the experiments five times, and averaged over the results.

As figure 4.3 shows, the "raw" feature vectors from section 3.2.3 are not very useful for this algorithm. Because they contain a very limited number of features, the distances between data instances are very coarse. This does not allow us to measure the classifier's confidence reliably. In particular, the classifier tends to return many metonymical readings. This has a positive effect on recall, particularly at the end of the learning curve, when more informative instances are added. The effect on precision, however, is more dramatic, and as a result, F-score goes up by a mere 2%. Its highest value of 42% is comparable with the initial result in the previous section — which was reached with 10% of the training data — and lies 20% below the maximum performance possible. Hence, grammatical features do not provide the semi-supervised algorithm with enough information.

In order to make the algorithm more successful, we need to measure TiMBL's confi-

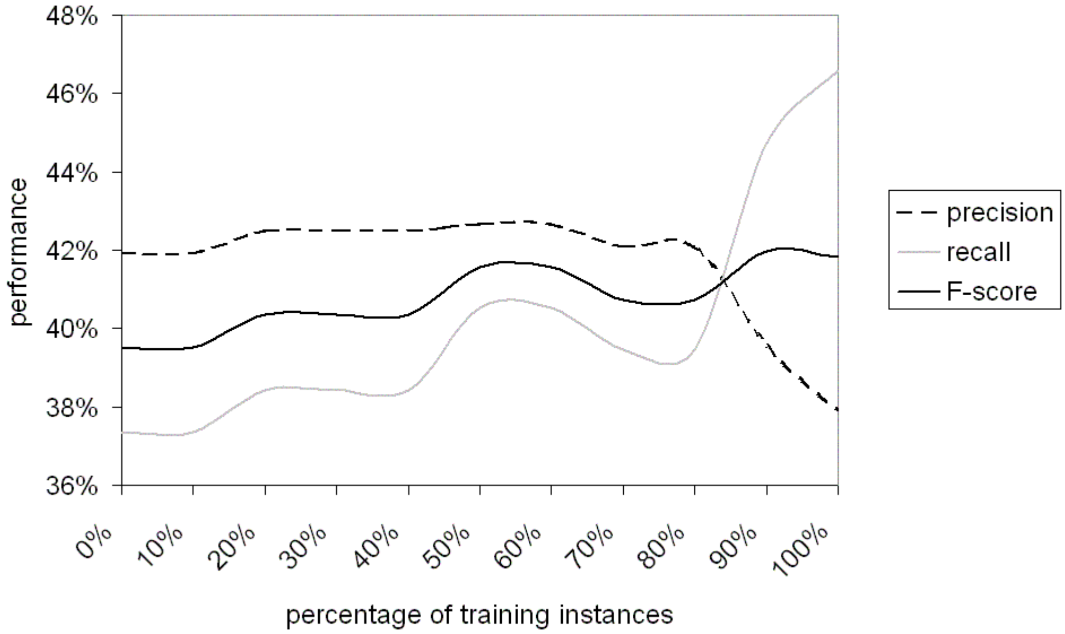

Figure 4.3: The semi-supervised learning curve for the country metonymies with grammatical features.

dence more reliably. This is made possible by the introduction of many more features, which lead to finer distances. I therefore included all WordNet hypernyms in the feature vectors. As figure 4.4 indicates, this indeed results in a very promising performance on the country data. While the effect on precision is not spectacular, recall goes up by 14%. This is a very good result: it shows that the classifier is able to return more metonymical labels without sacrificing precision. As a result, the F-score increases by 7%, and reaches 53% on average. This is only about 10% lower than the maximum F-score in the previous section, which relied on many more labelled training instances than the current ten.

The Hungary data in figure 4.5 gives less clear-cut results. Recall ends slightly higher, precision ends lower, and F-score stays about the same. This is probably due to the fact that there is less variety in the feature vectors of the Hungary data. As with the learning curves in the previous section, the maximum scores were reached with a training set of about 200 instances. After this point, little useful information is added to the training

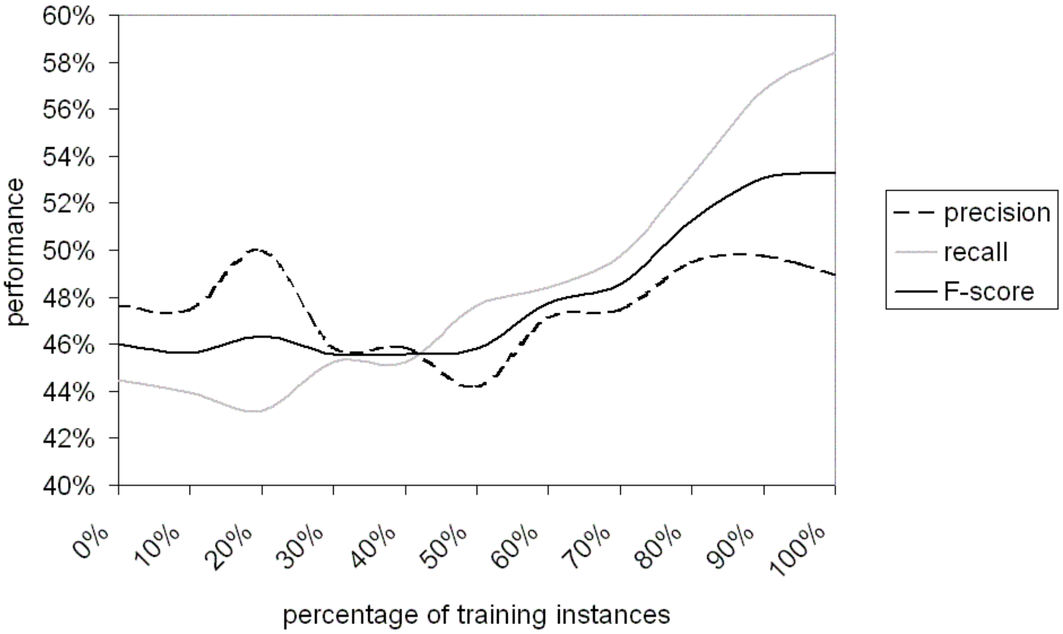

Figure 4.4: The semi-supervised learning curve for the country metonymies with grammatical and semantic features.

data. Nevertheless, the highest F-score lies above 50% (vs. 68% earlier), which is a promising result, given the fact that we only started with ten labelled instances.

In its present implementation, this semi-supervised algorithm has two major weaknesses. First, as the graphs above indicate, it starts by adding training instances that are rather uninformative. By always selecting the vectors with the smallest distance — and scores of exact matches — it adds those instances that most resemble the training data. While the classifier is most confident about those, they do not lead to much new knowledge. Changes in performance thus tend to show up in the second half of the graphs, when the classifier receives more additional information.

A better semi-supervised algorithm has to strike a balance between adding instances with a high confidence and a high informativity. Such techniques are studied in the field of Active Learning, which has proved its usefulness in NLP tasks such as parsing and Named Entity Recognition (see e.g. Hwa, 2002; Osborne and Baldridge, 2004). Successful Active Learning algorithms use a measure of certainty such as entropy and

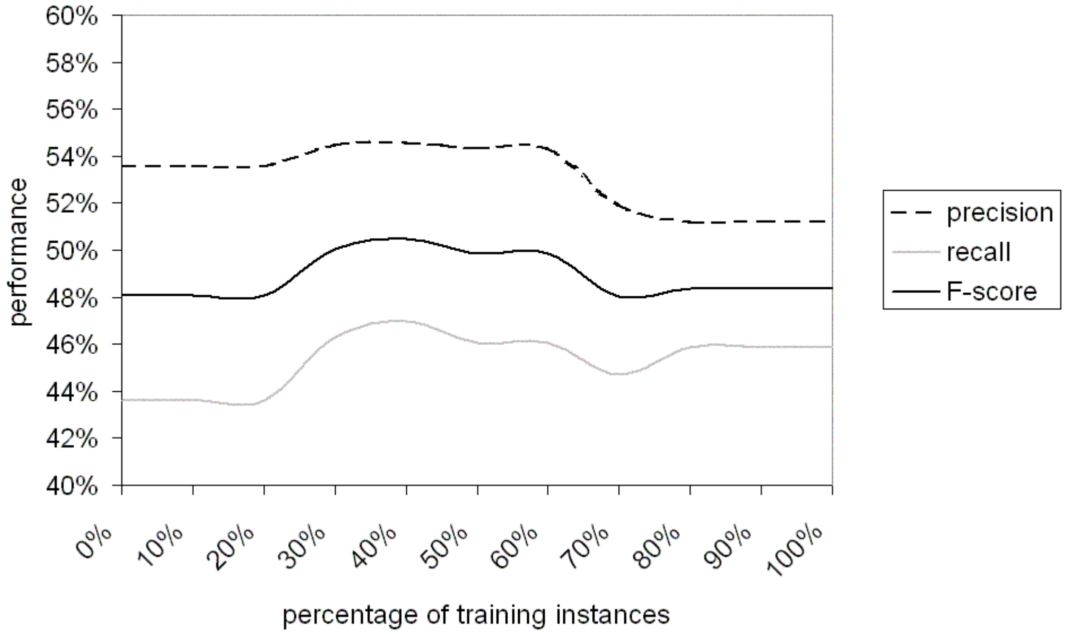

Figure 4.5: The semi-supervised learning curve for the Hungary metonymies with grammatical and semantic features.

therefore succeed in selecting new training instances with a high information gain. However, entropy relies on probabilities, and all TiMBL can offer us is the distance between two vectors. The implementation of an Active Learning algorithm for TiMBL is thus less than straightforward. Other learners, such as Naive Bayes classifiers, do return probabilities, and are therefore more appropriate for such an implementation.

The second weakness of the algorithm is its random selection of seed instances. If a very marginal seed is chosen for one or more of the target readings, this would obviously have the algorithm start off on the wrong track, and subsequently compromise performance. A better algorithm thus requires the selection of prototypical instances for each of the classes, either with or without human participation (see e.g. Hearst, 1991; Yarowsky, 1995). Future research is needed to see how this selection can proceed, and how it affects the results.

If these disadvantages are addressed, a semi-supervised algorithm may very well be able to return F-scores that rival those of its supervised competitors. The promising performance of the simple algorithm in this chapter certainly points in this direction.

4.4 Discussion

The figures above indicate that a semi-supervised algorithm may be used to address the knowledge acquisition bottleneck in metonymy recognition. My results proved that about half of the labelled country data with semantic information gave the same performance on the metonymical class as the entire training set without semantic classes. Similarly, the learning curve for the Hungary data indicated that a classifier with a large training set in fact gives a lower performance than one with only 200 training instances.

In addition, a simple semi-supervised algorithm was able to capitalize on the information contained in just ten labelled training examples. It improved initial recall on the country metonymies by 14%, without sacrificing performance. This resulted in an F-score of more than 53%. The performance gain on the Hungary set was more modest, but here F-score climbed above 50% as well. Future improvements such as the selection of prototypical seeds or the introduction of Active Learning techniques are certain to result in even more competitive semi-supervised algorithms.

Conclusions

Approach Metonymy is a figure of speech that uses "one entity to refer to another that is related to it" (Lakoff and Johnson, 1980, p.35). Although this relation can take on many forms, in practice it is possible to give a list of metonymical patterns to which most metonymies of a certain semantic class belong. Because of their ubiquity in everyday language, many NLP tasks need to be able to resolve metonymies correctly. Metonymy resolution involves two stages: metonymy recognition and interpretation.

Present approaches to metonymy recognition are very dependent on the construction of knowledge bases (Markert and Hahn, 2001) or the manual annotation of hundreds of training instances (Markert and Nissim, 2002a, 2005b; Nissim and Markert, 2003, 2005). This dissertation therefore investigated the possibility of developing knowledge-lean machine learning algorithms that considerably reduce the amount of human participation. The development of such algorithms is necessary for the generalization of current algorithms to a wide-scope metonymy resolution system. In order to address this issue, this thesis examined unsupervised, supervised and semi-supervised methods.

Contributions Chapter 2 approached metonymy recognition from the perspective of Schütze's (1998) unsupervised Word Sense Discrimination. This algorithm dispenses with human intervention altogether, but discriminates rather than disambiguates the senses of a target word. In its most basic implementation, which relies on co-occurrence information only, this technique did not prove very successful. However, when grammatical tags were added to the data and Singular Value Decomposition was dropped, it was much more useful. Six of the experiments with these properties were able to discover two sense clusters that were significantly correlated with the metonymical and literal readings of the data. This resulted in a maximum F-score of around 40% on the Hungary metonymies (see figure 4.6).

The robustness of these unsupervised algorithms is limited, however. Accuracy remained below the majority baseline, and the F-scores on the metonymical class were clearly inferior to the algorithms developed by Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005). Nevertheless, unsupervised algorithms may prove useful as a pre-processing step for the selection of data that have to be annotated. In this way they can contribute to the reduction of human effort in metonymy recognition systems.

The second stage of my research involved the development of a supervised algorithm that was able to replicate the results in Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005) in chapter 3. I chose to examine an algorithm that is much "lazier" than its competitors in metonymy recognition. Memory-based learning simply stores all training examples in its memory and classifies a test example by comparing it to the most similar training examples. I found that this system was able to rival Markert and Nissim's (2005b) most advanced `combination` algorithm, even without taking semantic information into account.

A few additional experiments investigated the benefit of semantic information and the effect of automatic grammatical annotation. Semantic information in the form of WordNet synsets did not increase performance. However, I later found that this may be due to the large number of training examples: with a smaller training set, semantic features have an extremely positive effect. Figure 4.6 shows that the supervised algorithm with WordNet information reached an F-score of about 68% when it was evaluated on a held-out test set in chapter 4. Finally, chapter 3 questioned the robustness of supervised systems by studying the effect of automatic grammatical annotation, which led to a drop in F-scores of 13% to 15%. Nevertheless, supervised algorithms outperform their unsupervised or semi-supervised variants.

The success of supervised algorithms, however, is compromised by their extensive

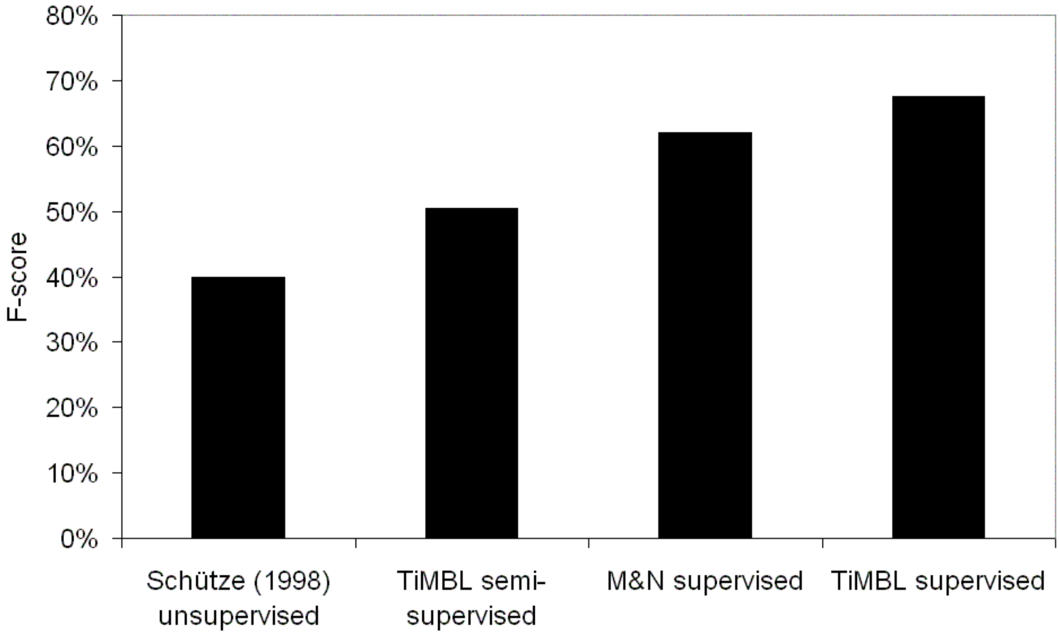

Figure 4.6: The F-scores on the Hungary metonymies of the three learning algorithms in this dissertation and Markert and Nissim's (2005b) (M&N) combination algorithm.

need for labelled training data. Chapter 4 therefore tried to strike a balance between unsupervised and supervised systems by examining a semi-supervised algorithm that only needs ten labelled training instances. It first showed that a subset of the labelled training data could lead to very good results. Possible reductions in training set size amounted to 50% for the country data, and even to 60% or 70% for the Hungary data.

The semi-supervised algorithm I implemented next gave some very promising results as well. It was able to increase the initial F-score on the country data by 7%, and reached maximum F-scores of above 50% on both test sets. This is an improvement of 10% over the unsupervised data — a promising result, given that only ten labelled instances were required.

In short, my exploration of three machine learning approaches to metonymy recognition has shown that it is possible to develop knowledge-lean algorithms that significantly reduce human participation. First, unsupervised algorithms can be used as a pre-processing step for the selection of training instances. Second, "lazy" supervised algorithms are able to replicate the results of more complex systems. It moreover proved possible to reduce the size of the training sets considerably. And finally, even though the performance of my semi-supervised algorithm was below that of its supervised competitors, it still leaves much room for future improvements.

Future work Many issues still remain to be addressed. In the previous chapter, I pointed out that the simple semi-supervised algorithm has a bias for uninformative instances — a problem that may be solved by Active Learning. Similarly, performance is likely to benefit from the selection of prototypical seeds for each of the target readings.

In chapter 3, I mentioned that reliable metonymy recognition needs to take more syntactic information into account. At the same time, it should be investigated how robust grammatical information can be obtained without manual annotation, since a parser such as RASP led to a dramatic drop in performance.

Remember, finally, that the approaches in this dissertation are intended to tackle metonymy recognition only, since they are not able to fully interpret the metonymical words. This also applies to the algorithms in Markert and Nissim (2002a, 2005b) and Nissim and Markert (2003, 2005). As I pointed out in chapter 1, these were only applied to the highest level of the hierarchy of metonymical patterns, and therefore do yet not offer a full interpretation. A complete metonymy resolution system should thus combine a metonymy recognition algorithm like the ones I presented with an interpretation algorithm such as that in Utiyama et al. (2000). So far, most systems only address one of the two related problems.

In conclusion, the future for metonymy recognition does not lie in the careful manual construction of ever-expanding knowledge bases, nor in the annotation of more and more data. Instead, it should be studied how knowledge-lean algorithms can be made to tackle a wider variety of target words and metonymical patterns, while their demand for human labour is kept low. This dissertation has taken the first steps along this path.

Bibliography

Briscoe, E. and Carroll, J. (2002). Robust accurate statistical annotation of general text. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, Las Palmas, Spain.

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

Cutting, D. R., Pedersen, J. O., Karger, D., and Tukey, J. W. (1992). Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Copenhagen, Denmark.

Daelemans, W., Zavrel, J., Van der Sloot, K., and Van den Bosch, A. (2004). TiMBL: Tilburg Memory-Based Learner. Technical report, Induction of Linguistic Knowledge, Computational Linguistics, Tilburg University.

Fass, D. (1997). *Processing Metaphor and Metonymy*. Stanford, CA: Ablex.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Gaustad, T. (2004). *Linguistic Knowledge and Word Sense Disambiguation*. PhD thesis, University of Groningen.

Golub, G. H. and Van Loan, C. F. (1989). *Matrix Computations*. London: The Johns Hopkins University Press.

Guthrie, J. A., Guthrie, L., Wilks, Y., and Aidinejad, H. (1991). Subject-dependent co-occurrence and word sense disambiguation. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL-91)*, Berkeley, USA.

Hearst, M. A. (1991). Noun homograph disambiguation. In *Proceedings of the 7th Annual Conference of the University of Waterloo Centre for the New OED and Text Research*, Oxford, UK.

Hwa, R. (2002). Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276.

Jurafsky, D. and Martin, J. H. (2000). *Speech and Language Processing*. Upper Saddle River, NJ: Prentice Hall.

Kövecses, Z. (2002). *Metaphor: A Practical Introduction*. Oxford: Oxford University Press.

Kövecses, Z. and Radden, R. (1998). Metonymy: Developing a cognitive linguistic view. *Cognitive Linguistics*, 9(1):37–77.

Kulkarni, A. and Pedersen, T. (2005). SenseClusters: Unsupervised clustering and labeling of similar contexts. In *Proceedings of the Demonstration and Interactive Poster Session of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, USA.

Lakoff, G. and Johnson, M. (1980). *Metaphors We Live By*. London: The University of Chicago Press.

Landauer, T. K., Laham, D., Rehder, R., and Schreiner, M. E. (1997). How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. In *Proceedings of the 19th Annual Conference of the Cognitive Science Society*, Mahwah, USA.

Lapata, M. and Lascarides, A. (2003). A probabilistic account of logical metonymy. *Computational Linguistics*, 29(2):261–312.

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

Lesk, M. E. (1986). Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the Fifth International Conference on Systems Documentation*, Toronto, Canada.

Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the International Conference on Machine Learning*, Madison, USA.

Lund, K., Burgess, C., and Atchley, R. A. (1995). Semantic and associative priming in high-dimensional semantic space. In *Proceedings of the 17th Annual Conference of the Cognitive Science Society*, Hillsdale, USA.

Markert, K. and Hahn, U. (2001). Understanding metonymies in discourse. *Artificial Intelligence*, 135(1/2):145–198.

Markert, K. and Nissim, M. (2002a). Metonymy resolution as a classification task. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, Philadelphia, USA.

Markert, K. and Nissim, M. (2002b). Towards a corpus annotated for metonymies: the case of location names. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*, Las Palmas, Spain.

Markert, K. and Nissim, M. (2005a). Annotation scheme for metonymies. Technical report, University of Edinburgh, available from `www.cogsci.ed.ac.uk/~malvi/mascara/publications.html`.

Markert, K. and Nissim, M. (2005b). Metonymies in-the-large: Robust and scalable metonymy resolution. Unpublished final report.

Nissim, M. and Markert, K. (2003). Syntactic features and word similarity for supervised metonymy resolution. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, Sapporo, Japan.

Nissim, M. and Markert, K. (2005). Learning to buy a Renault and talk to BMW: A supervised approach to conventional metonymy. In Bunt, H., editor, *Proceedings of the 6th International Workshop on Computational Semantics*, Tilburg, The Netherlands.

Norrick, N. R. (1981). *Semiotic Principles in Semantic Theory*. Amsterdam Studies in the Theory and History of Linguistic Science IV. Current Issues in Linguistic Theory, Volume 20. Amsterdam: John Benjamins.

Nunberg, G. (1978). *The Pragmatics of Reference*. PhD thesis, City University of New York.

Osborne, M. and Baldridge, J. (2004). Ensemble-based active learning for parse selection. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL)*. Boston, USA.

Peirsman, Y. and Geeraerts, D. (acc.). Metonymy as a prototypical category. *Cognitive Linguistics*.

Purandare, A. and Pedersen, T. (2004a). SenseClusters — finding clusters that represent word senses. In *Proceedings of the Fifth Annual Meeting of the North American Chapter of the Association for Computational Linguistics*, Boston, USA.

Purandare, A. and Pedersen, T. (2004b). Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the Conference on Computational Natural Language Learning*, Boston, USA.

Pustejovsky, J. (1995). *The Generative Lexicon*. Cambridge, MA: MIT Press.

Sahlgren, M. (2002). Towards a flexible model of word meaning. Acquiring (and Using) Linguistic (and World) Knowledge for Information Access, AAAI Spring Symposium, Stanford University, Palo Alto, USA.

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

Utiyama, M., Murata, M., and Isahara, H. (2000). A statistical approach to the processing of metonymy. In *Proceedings of the 18th International Conference on Computational Linguistics*, Saarbrücken, Germany.

Wiemer-Hastings, P. and Zipitria, I. (2001). Rules for syntax, vectors for semantics.

In *Proceedings of the 23rd Annual Conference of the Cognitive Science Society*, Mahwah, USA.

Yarowsky, D. (1995). Unsupervised word sense disambiguation rivaling supervised mehods. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-95)*, Cambridge, USA.

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