Word Sense Disambiguation Based on Sense Similarity and Syntactic Context

by

Basak Mutlum

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science
in
Computer Engineering

Koc University
September 2005

Koc University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Basak Mutlum

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Deniz Yuret, Ph. D. (Advisor)

_____

Engin Erzin, Ph. D.

_____

Aylin Kuntay, Ph. D.

Date: _____

ABSTRACT

Word Sense Disambiguation (WSD) is the task of determining the meaning of an ambiguous word within a given context. It is an open problem that has to be solved effectively in order to meet the needs of other natural language processing tasks. Supervised and unsupervised algorithms have been tried throughout the WSD research history. Up to now, supervised systems achieved the best accuracies. However, these systems with the first sense heuristic have come to a natural limit. In order to make improvement in WSD, benefits of unsupervised systems should be examined.

In this thesis, an unsupervised algorithm based on sense similarity and syntactic context is presented. The algorithm relies on the intuition that two different words are likely to have similar meanings if they occur in similar local contexts. With the help of a principle-based broad coverage parser, a 100-million-word training corpus is parsed and local context features are extracted based on some rules. Similarity values between the ambiguous word and the words that occurred in a similar local context as the ambiguous word are evaluated. Based on a similarity maximization algorithm, polysemous words are disambiguated. The performance of the algorithm is tested on SENSEVAL-2 and SENSEVAL-3 English all-words task data and an accuracy of 59% is obtained.

To my mum…

TABLE OF CONTENTS

List of Tables **vii**

List of Figures **viii**

Chapter 1: Introduction **1**

Chapter 2: Word Sense Disambiguation **5**

2.1 WSD Approaches. 7

2.1.1 Supervised Disambiguation. 7

2.1.2 Unsupervised Disambiguation. 11

2.2 Knowledge Bases for WSD. 12

2.2.1 Machine Readable Dictionaries. 12

2.2.2 Thesauri 13

2.2.3 Computational Lexicons 14

2.3 Information Sources for WSD. 18

2.4 General Approach and Related Work 21

Chapter 3: Word Sense Disambiguation Based on Sense Similarity and Syntactic Context **24**

3.1 Introduction. 24

3.2 Resources of the Algorithm 27

3.2.1 Training Corpus. 28
3.2.2 Concept Hierarchy. 28
3.2.3 Similarity Measure. 28
3.2.4 Broad-Coverage Parser 33
3.3 Local Context 35
3.4 Local Context Feature Database. 38
3.5 Disambiguation Algorithm. 45
3.6 Evaluation and Results 52

Chapter 4: Conclusion and Future Work **55**

Bibliography **58**

Vita **69**

LIST OF TABLES

Table 3.1: Grammatical Categories of MINIPAR 34
Table 3.2: Grammatical Relationships of MINIPAR 35
Table 3.3: Results of Senseval data 52
Table 3.4: Top-10 performing systems in SENSEVAL-2 53
Table 3.5: Comparison of similarity measures 54

LIST OF FIGURES

Figure 2.1: WordNet 2.0 entry for the noun "*car*" 15

Figure 2.2: Noun relations in WordNet 16

Figure 2.3: Verb relations in WordNet 16

Figure 2.4: Adjective and adverb relations in WordNet 16

Figure 2.5: Hyponymy chain for the noun "*valley*" 17

Figure 3.1: Words that have a relation with *"apply"* via *"for"* 26

Figure 3.2: A fragment of WordNet 32

Figure 3.3: Examples of dependencies generated by MINIPAR 37

Figure 3.4: Examples of local context feature database entries 45

Figure 3.5: Similarity matrix 48

Figure 3.6: Words that have a relationship with *"world"* via *"of"* 49

Figure 3.7: WordNet 2.0 entry for the noun *"work"* 50

Figure 3.8: Hypernyms of the noun *"work"* 51

Chapter 1

INTRODUCTION

Natural language is an integral part of our lives. Language serves as the primary vehicle by which people communicate and record information. It has the potential for expressing an enormous range of ideas, and for conveying complex thoughts. Because it is so integral to our lives, we usually take its powers and influence for granted.

The aim of computational linguistics is, in a sense, to capture this power. It tries to enable computers to be used as aids in analyzing and processing natural language, and to understand, by analogy with computers, more about how people process natural language. By understanding language processes in procedural terms, we can give computer systems the ability to generate and interpret natural language. This would make it possible for computers to perform linguistic tasks (such as translation), process textual data (books, journals, newspapers), and make it much easier for people to access computer-stored data.

In the field of computational linguistics, some results have already been obtained however, a number of important research problems have not been solved yet. Ambiguity is one of these problems which have been a great challenge for computational linguists. In general, people are unaware of the ambiguities in the language they use because they are very good at resolving them using context and their knowledge of the world. But computer systems do not have this knowledge, and consequently do not do a good job of making use of the context.

Something is ambiguous when it can be understood in two or more possible ways or when it has more than one meaning. If the ambiguity is in a sentence or clause, it is called structural (syntactic) ambiguity. If it is in a single word, it is called lexical ambiguity.

For the structural ambiguity, consider the sentence "The man saw the girl with the telescope". This sentence is ambiguous since it can be interpreted in two ways: The man saw the girl who possessed the telescope or, the man saw the girl with the aid of the telescope. However, the sentence "The man saw the girl with a red hat" is not ambiguous for a human reader (people have the knowledge that a hat cannot be used to see), while it has the same ambiguity as the previous example for a computer.

Lexical semantic ambiguity occurs when a single word is associated with multiple senses. In this thesis, we will be focusing on lexical semantic ambiguity. Examples of lexical ambiguity are everywhere. In fact, almost any word has more than one meaning. For example, consider the noun *party*. It can refer to at least 5 different things as follows:

- an organization to gain political power
- an occasion on which people can assemble for social interaction and entertainment
- a band of people associated temporarily in some activity
- a group of people gathered together for pleasure
- a person involved in legal proceedings

All of the above senses can be subsumed in just one sense such as "group of people", however, for various applications, such as information retrieval or machine translation, it is important to be able to distinguish between the different senses of a word. In a machine translation application, different senses of a word may be represented with different words in the target language. In order to correctly translate a text in one language to another, firstly we have to know the senses of the words and then find the best translation equivalent in the target language. Apart from these, for many other words there is no such general sense like the one for the noun *party*.

Lexical ambiguity can refer to both homonymy and polysemy. Homonyms are words that are written the same way, but are (historically or conceptually) really two different words with different meanings which seem unrelated. Examples are *suit* ("lawsuit" and "set of garments") and *bank* ("river bank" and "financial institution"). If a word's meanings are related, it is called a polyseme. The word *party* is polysemous because its senses can be generalized as "group of people", that is they are related.

Now let us consider the meaning of the noun *party* in the following sentence:

Mr. Smith's party took 38% of the votes in the last election. (1.1)

It is clear to a human reader that the noun *party* is in the sense "an organization to gain political power" in the sentence (1.1). Most people are not even aware of the ambiguity contained in the sentence. Humans are so skilled at resolving potential ambiguities that they do not realize they are doing it. There has been research on how people resolve ambiguities; however we still do not know exactly how humans do lexical disambiguation. Therefore, it is a difficult task to teach a computer to do the same thing. If there is more than one ambiguous word in a sentence, the number of potential interpretations of the sentence increases dramatically. The number of interpretations of a sentence is the product of all possible meanings of the words that construct that sentence. For the sentence above, *party* has 5, *take* has 42, *vote* has 5, *last* has 10, and *election* has 4 senses[1]. Therefore there are 42000 possible interpretations for the example sentence. The most prominent way to disambiguate a word is examining its context. The context can be considered as the words surrounding the ambiguous word, which is the noun *party* in our case. Words as *vote* and *election* might be a good clue for the sense of the noun *party*. But context is not the only information available for disambiguation. Syntactic classes of the words in the ambiguous

[1] According to Wordnet 2.0.

word's context (whether they are noun, verb or adjective, etc.), whether the ambiguous word plays the role of object or subject in the syntactic structure of the sentence may also be used in the disambiguation process.

The main research question in this thesis is determining the meaning of a word in a particular usage, namely Word Sense Disambiguation, by using sense similarity and syntactic context. In order to achieve this, we make use of a WSD algorithm that relies on the intuition that "two different words are likely to have similar meanings if they occur in similar local contexts".

This thesis is constructed as follows. An overview of the problem of Word Sense Disambiguation and a literature survey is presented in Chapter-2. Chapter-3 contains the description of our WSD system and presents the results obtained from this system on SENSEVAL-2 and SENSEVAL-3 English all-words data [1, 2, 3]. Finally, we conclude in Chapter-4 with some final remarks on the findings presented in this thesis.

Chapter 2

WORD SENSE DISAMBIGUATION

Many words have more than one possible meaning, for example a bat can be a small nocturnal creature or a piece of sports equipment and a bank can mean the edge of a river or a financial institution that accepts money. When we look up a word in any dictionary, it can be seen that a word may have many meanings some of which are very different from each other. In fact in some cases a word can have two meanings which are the opposite of one another such as *"consult"* which can mean to ask for advice or give advice and *"clip"* which can mean to fasten or detach. Given these complications, it is important for a computer to correctly determine the meaning in which a word is used.

Word Sense Disambiguation (WSD) refers to the resolution of lexical semantic ambiguity and its goal is to attribute the correct senses to words in a given context. It has been described as AI-complete, which is used to describe problems or subproblems in AI, to indicate that the solution presupposes a solution to the "strong AI problem" (that is, the synthesis of a human-level intelligence). A problem that is AI-complete is, in other words, just too hard.

WSD is regarded as one of the most interesting and longest-standing problems in natural language processing. There are many uses for WSD. The most obvious application of WSD is Machine Translation. The machine translation process requires at least two stages: understanding the source language and generating sentences in the target language. WSD is required in both stages since a word in the source language may have more than one possible translation in the target language. For example, the English word *"drug"* can

be translated into Turkish as *"ilaç"* for its sense of *"medicine"* or as *"uyuşturucu"* for its sense of *"dope"* depending on the context. In order to be able to correctly translate a text, we need to know which sense is intended in the text. However, a recently debated question in the conferences is whether even the new state of-the-art word sense disambiguation models actually have anything to offer to full statistical machine translation systems. Carpuat and Wu [4] made some experiments using a WSD system based on the model that achieved the best performance on the Senseval-3 Chinese lexical sample task (Carpuat *et al.*, 2004) [5]. They argue empirically that their statistical machine translation model is sufficiently accurate so that within the training domain, even the state-of-the-art WSD model they used is only able to improve on its lexical choice predictions in a relatively small proportion of cases. They also argue that even when the dedicated WSD model makes better predictions, current statistical machine translation models are unable to exploit this.

Information Retrieval also benefits from WSD. Ambiguous words in the queries are problematic for information retrieval systems. There are two types of benefits: First one is, filtering out bad keyword matches; i.e. same word can be used in different senses in questions and answers. Secondly, WSD can be used in finding matches that use different words, i.e. finding the synonym of the question word in the answer. Hence, retrieval engines need WSD for finding out documents with senses relevant to the query.

Another potential area for WSD is Speech Processing. In speech synthesis, it is important to determine the correct pronunciations of words in order to generate speech that sounds natural. This process is difficult since there exists some words which are pronounced in more than one way depending on their meaning. For example, as mentioned in Stevenson 2003 [6], *"lead"* is pronounced in one way when it is used in the sense "be in front" and in another way when it is used in the sense *"a type of metal"*. WSD could help speech synthesis by identifying the correct sense of the word which will also provide the

correct pronunciation. The reverse problem may occur in speech recognition for homophones, words that are spelled differently but pronounced in the same way. If different spellings are treated as different senses, WSD can also be helpful in this situation.

Applications in text processing, grammatical analysis, content and thematic analysis also benefit from WSD techniques.

In the following section, we will proceed with defining WSD approaches based on the different types of training material and an overall literature review on WSD.

2.1 WSD Approaches

WSD algorithms can be divided into two based on the corpora used for training. These approaches are:

i. Supervised Word Sense Disambiguation

ii. Unsupervised Word Sense Disambiguation

In supervised WSD the training data is sense-tagged whereas in unsupervised WSD the training data is raw corpora which have not been semantically disambiguated. In the following sections these approaches will be explained in detail.

2.1.1 Supervised Disambiguation

Supervised disambiguation is an application of the supervised learning approach for creating a classifier. A disambiguated corpus where each occurrence of an ambiguous word is annotated with a contextually appropriate sense is available for training. The aim in supervised disambiguation is to build a classifier which correctly classifies new cases based on their context of use.

Machine learning algorithms such as Bayesian classifiers (Duda and Hat, 1973) [7], decision lists (Rivest, 1987) [8], decision trees (Quinlan, 1986) [9], k-nearest neighbor and neural networks (Rumelhart et al., 1986) [10] are examples of supervised learning algorithms.

An example of probabilistic algorithms is Naïve Bayes (Mitchell 1997:ch. 6) [11] which has been frequently applied in WSD with good results (Pedersen, 2000) [12]. Gale, Church and Yarowsky [13;14] uses a variant of Bayes ratio on six ambiguous nouns, namely *drug*, *duty*, *land*, *language*, *position*, and *sentence*, and reports 90% accuracy in discriminating between two senses of these words. Mooney [15] reports that Naïve Bayes and neural networks achieved the highest performance with an accuracy of 73% in assigning the correct senses to a corpus of examples of word *line* which has six senses. The other algorithms in Mooney's survey were 3-nearest neighbors, perceptron, decision tree, decision list and logic programming variants. Combining various classifiers has also been tested. Florian et al. [16] combined four classifiers namely feature-enhanced Naïve Bayes, Cosine, bag-of-words Naïve Bayes and non-hierarchical decision lists.

Decision lists search for discriminatory features in the training corpus and build a set of rules for disambiguation. Yarowsky [17] makes use of hierarchical decision lists and achieves top performance in the SENSEVAL-1 framework on the 36 test words for which tagged training data was available. Agirre and Martinez [18] reports that decision lists provide state-of-the-art results with simple and very fast means. This approach is reported to learn with low amounts of data.

Decision lists and Bayesian classifiers are the most popular algorithms in supervised disambiguation. For neural networks Towell and Voorhees [19], for decision trees Black [20] and Pedersen [21], for k-nearest neighbor Ng and Lee [22] and for information-theoretic approaches Brown et al. [23] are some examples of the work done on WSD.

A major problem with supervised approaches is the need for a large sense-tagged training set. Despite the availability of large corpora, manually sense-tagging of a corpus is very difficult and very few sense-tagged data are available now.

The two largest corpora that are available are the SemCor corpus (Landes et al., 1998) [24] and the SENSEVAL corpus (Kilgariff and Rosenzweig, 2000)[2]. The SemCor corpus, created by the Princeton University, is a subset of the English Brown corpus containing almost 700,000 running words. In SemCor, all the words are tagged by part of speech and more than 200,000 content words are also lemmatized and sense-tagged according to Princeton WordNet 1.6 (mappings for later versions of WordNet are also available). SENSEVAL corpus is derived from the HECTOR corpus and dictionary project. It is a joint Oxford University Press and Digital project which took place in the early 1990s. Another sense-tagged corpus available is the DSO Corpus of Sense-Tagged English (Ng and Lee, 1996) [22]. This corpus contains sense-tagged word occurrences for 121 nouns and 70 verbs which are among the most frequently occurring and ambiguous words in English. These occurrences are provided in about 192,800 sentences taken from the Brown corpus and the Wall Street Journal and have been hand tagged by students at the Linguistics Program of the National University of Singapore. WordNet 1.5 sense definitions of these nouns and verbs were used to identify a word sense for each occurrence of each word.

There have been several efforts for finding a way to avoid the use of hand-tagged data. ***Bootstrapping*** is the most frequently used method for this purpose. Bootstrapping relies on a small number of instances of each sense for each lexeme of interest. These sense-tagged instances are used as seeds to train an initial classifier. This initial classifier is then used to extract a larger training set from the remaining untagged corpus. With each iteration of this process, the training corpus grows and the untagged corpus shrinks.

Hearst [25] generates a seed set by simply hand-tagging a small set of examples from the untagged corpus. However, during the training phase each occurrence of a set of nouns to be disambiguated is manually sense-tagged in several occurrences. Schütze [26; 27] proposes a method that avoids tagging each occurrence in the training corpus. Yarowsky [28] proposes an alternative technique by using two constraints named as "One sense per collocation" and "One sense per discourse" and reports an accuracy of 96% on twelve words. "One sense per collocation" argues that nearby words provide strong and consistent clues to the sense of a target word, conditional on relative distance, order and syntactic relationship. Also, "One sense per discourse" constraint argues that the sense of a target word is highly consistent within any given document. Different bootstrapping techniques are also presented in Mihalcea and Moldovan [29] and Mihalcea [30]. Mihalcea [30] makes a comparison between the results when training is performed on hand-tagged data and the results when training is done using the generated corpus by bootstrapping. She reports that the precision achieved with the generated corpus is comparable, and sometimes better than the precision achieved with hand-tagged corpora.

Another method for avoiding hand-tagged data is using ***parallel corpora*** [31]. In this method, bilingual corpora are used since different senses of some words translate differently in another language. By using a parallel aligned corpus, the translation of each occurrence of such words can be used to determine their correct senses automatically. In Dagan and Itai [32], Ide et al. [33] and Ng et al. [34], various uses of parallel corpora for WSD and its disadvantages can be found.

The main problem that supervised disambiguation methods face with is data sparseness. Since the sense-tagged training corpus is finite and very small for WSD, some senses of polysemous words are very likely to be missing and most of them have few examples. For a supervised algorithm to be successful, the training data must ensure that all senses of a polysemous word are covered. ***Smoothing*** is used to solve the data sparseness problem.

The task of reevaluating some of the zero-probabilities or low-probabilities and assigning them non-zero values is called smoothing. Some of the smoothing methods are add-one smoothing, Witten-Bell smoothing [35], and Good-Turing smoothing [36]. Gale [37], presented a Good-Turing method for estimating the probabilities of seen and unseen objects in linguistic applications named as Simple Good-Turing method.

2.1.2 Unsupervised Disambiguation

In machine learning the distinction between supervised and unsupervised algorithms rests on whether a set of classifications exists. In unsupervised word sense disambiguation, information is gathered from raw corpora which have not been semantically disambiguated.

Yarowsky [38] proposed an approach for marking words with their categories from a thesaurus. He used Roget's Thesaurus (Chapman, 1977) [39]. Training was carried out on an untagged corpus of 10 million words obtained from the electronic version of the Grollier's Encyclopedia. The important aspect of the approach was that he used a context of 50 words either side so that 100 words were considered in the training examples for each ambiguous word. This method was tested on 12 ambiguous words and reported to achieve 92% accuracy. Yarowsky notes that this method is best for extracting topical information, most successful for nouns. The algorithm presented in Yarowsky [28] is also an unsupervised algorithm making use of a bootstrapping procedure.

McCarthy et al. [40], presents an algorithm that makes use of a thesaurus acquired from raw textual corpora and the WordNet similarity package to find predominant noun senses automatically. The acquired predominant senses gave a precision of 64% on the nouns of the SENSEVAL-2 all-words task which is a promising result regarding that no hand-tagged data is used.

Some of the unsupervised methods correspond to clustering tasks rather than sense tagging tasks because they do not label words to predefined senses. These algorithms do not make use of an outside source of knowledge to define senses. This is called *Word Sense Discrimination* rather than disambiguation. They divide the occurrences of a word into a number of classes by determining for any two occurrences whether they belong to the same sense or not (Schütze and Pedersen 1995; Pedersen and Bruce, 1997; Schütze, 1998)[41, 42, 43]. Schütze's [43] results indicate that for coarse binary distinctions, unsupervised techniques can achieve results approaching those of supervised and bootstrapping methods. Purandere and Pedersen [44] present a systematic comparison of discrimination techniques proposed by Pedersen and Bruce [42, 45] and by Schütze [26, 43].

Infrequent senses and senses that have few collocations are hard to isolate in unsupervised disambiguation.

2.2 Knowledge Bases for WSD

In this section, different kinds of knowledge bases are presented. These knowledge bases can be used in any WSD system, whether it is supervised or unsupervised.

2.2.1 Machine Readable Dictionaries

Machine readable dictionaries (MRD) provide a ready-made information source of word senses. The first attempt to use MRD's came from Lesk (1986)[46]. He starts from the simple idea that a word's dictionary definitions are likely to be good indicators of the senses they define. By using Oxford Advanced Learner's Dictionary (OALD), he counts overlapping content words in the sense definitions of the ambiguous word and in the definitions of context words occurring nearby and selects the sense that achieves the

maximum number of overlaps. The accuracy of the method is reported to be 50-70% on short samples of the Jane Austen novel Pride and Prejudice and an Associated Press news story based on very brief experimentation with the program.

Cowie et al. [47] tried to improve Lesk's approach by optimizing the overlap of all words in a single sentence simultaneously. However, it was found computationally very expensive. Therefore, Cowie et al. [47] used simulated annealing[48] for the first time in natural language processing. They evaluated this approach using a corpus consisting of 50 example sentences taken from Longman Dictionary of Contemporary English (LDOCE) which were disambiguated by hand. 47% of the words were reported to be correctly disambiguated to sense level and 72% to more rough grained senses.

Stevenson and Wilks [49] computed the overlap by normalizing the contribution of a word to the overlap count. Pedersen and Banerjee [50] described a different version of the Lesk's algorithm by employing glosses contained in WordNet [24, 51], a lexical database for English which will be described later in detail.

Because of the fact that dictionaries are created for human use, not for computers, there are some inconsistencies (Veronis and Ide 1991; Ide and Veronis 1993a, 1993b)[52, 53, 54]. Although they provide detailed information at the lexical level, they lack pragmatic information used for sense determination. For instance, the relation between *ash* and *tobacco*, *cigarette* or *tray* is very indirect in a dictionary whereas the word *ash* co-occurs very frequently with these words in a corpus [55].

2.2.2 Thesauri

Thesauri provide information about relationships among words. Thesaurus based disambiguation makes use of the semantic categorization provided by a thesaurus or a dictionary with subject categories. The most frequently used thesaurus in WSD is Roget's

International Thesaurus (Roget, 1946) which was put into machine-tractable form in 1950's [39].

Walker [56] proposed an algorithm as follows: each word is assigned to one or more subject categories in the thesaurus. If the word is assigned to several subjects, then it is assumed that they correspond to different senses of the word.

Similar to machine readable dictionaries, a thesaurus is a resource for humans, so there is not enough information about word relations.

2.2.3 Computational Lexicons

The usefulness of lexical relations in linguistic, psycholinguistic and computational research has led to a number of efforts to create large electronic databases of such relations. Beginning from the mid-1980's, construction of semantic lexicons by hand has emerged. Some examples of these lexicons are WordNet [24, 51], CyC [57], ACQUILEX [58], and COMLEX [59]. Each of these lexicons contains different kinds of information.

Since WordNet is the most popular lexicon among the above and since it is used in the work done in this thesis both for sense evaluation and for similarity measure, this section presents detailed information about it.

WordNet

WordNet is an online lexical reference system which was developed at Princeton University under the direction of Professor George A. Miller. It combines many features used for WSD in one system. It includes definitions of word senses as in a dictionary; it defines "synsets" of synonymous words representing a single lexical concept; and it includes word-to-word relations.

WordNet consists of three databases: noun database, verb database and one database for adjectives and adverbs. Each database consists of lexical entries corresponding to unique orthographic forms. Each form is associated with a set of senses.

Overview of noun car

The noun car has 5 senses (first 3 from tagged texts)

1. (598) car, auto, automobile, machine, motorcar -- (4-wheeled motor vehicle; usually propelled by an internal combustion engine; "he needs a car to get to work")
2. (24) car, railcar, railway car, railroad car -- (a wheeled vehicle adapted to the rails of railroad; "three cars had jumped the rails")
3. (1) cable car, car -- (a conveyance for passengers or freight on a cable railway; "they took a cable car to the top of the mountain")
4. car, gondola -- (car suspended from an airship and carrying personnel and cargo and power plant)
5. car, elevator car -- (where passengers ride up and down; "the car was on the top floor")

Figure 2.1: WordNet 2.0 entry for the noun "*car*"

WordNet's sense entries consist of a set of synonyms, a dictionary-style definition, or gloss, and some example uses as shown in Figure 2.1. The numbers in front of some senses are the frequency values obtained from SemCor corpus. This simple listing of lexical entries looks like an ordinary dictionary. However, WordNet contains a set of domain-independent lexical relations which hold among synsets or lemmas, i.e. lexical or semantic pointers. Figure 2.2, 2.3 and 2.4 show a subset of the relations associated with each of the three databases.

Relation	Definition	Example
Hypernym	From concepts to superordinates	breakfast → meal
Hyponym	From concepts to subtypes	meal → lunch
Has-member	From groups to their members	faculty → professor
Member-of	From members to their groups	pilot → crew
Has-part	From wholes to parts	table → leg
Part-of	From parts to wholes	course → meal
Antonym	Opposites	leader → follower

Figure 2.2: Noun relations in WordNet

Relation	Definition	Example
Hypernym	From events to superordinate events	fly → travel
Troponym	From events to their subtypes	walk → stroll
Entails	From events to the events they entail	snore → sleep
Antonym	Opposites	increase → decrease

Figure 2.3: Verb relations in WordNet

Relation	Definition	Example
Antonym (adjective)	Opposite	heavy → light
Antonym (adverb)	Opposite	quickly → slowly

Figure 2.4: Adjective and adverb relations in WordNet

Synonym and hyponym relations are the most frequently used relations in WordNet. Synonym means different lemmas with the same meaning. Two WordNet lemmas are considered to be synonyms if they can be successfully substituted in some context. A synset is a set of synonyms. Consider the following example of a synset:

{ car, auto, automobile, machine, motorcar }

The gloss of this synset describes it as “4-wheeled motor vehicle; usually propelled by an internal combustion engine; "he needs a car to get to work" as shown in Figure 2.1 for

the first sense of noun *car*. Instead of representing concepts using logical terms, WordNet represents them as synsets.

Each noun synset is related to its immediately more general and more specific synsets via direct hypernym and hyponym relations. To find chains of more general or more specific synsets, one can follow a transitive chain of hypernym and hyponym relations. As an example, the hypernym chain for the noun *valley* is shown in Figure 2.5. In this figure, successively more general synsets are shown on successive indented lines. The chain starts from *valley* and ends in *entity*.

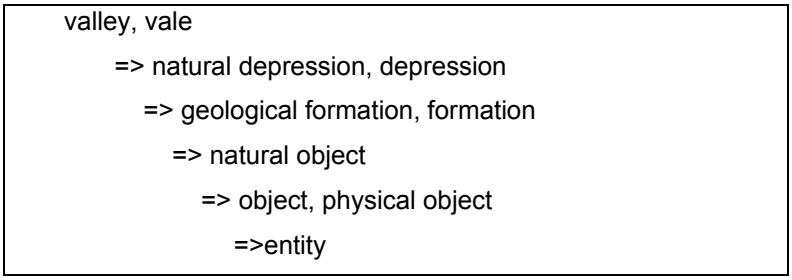


Figure 2.5: Hyponymy chain for the noun "*valley*"

Since WordNet provides the broadest set of lexical information in one resource, it is used frequently. Another reason for its wide-spread use is, it is the first broad-coverage lexical resource that is freely available.

The earliest attempts to use WordNet in WSD were in information retrieval field. Voorhees [60] and Richardson and Smeaton [61] created knowledge bases using WordNet's hierarchy. Li et al. [62] proposed a WordNet-based algorithm for WSD. Disambiguation was done by semantic similarity between words and heuristic rules. Heuristic rules were based on the semantic similarity and the WordNet hierarchy. Leacock et al. [63] used WordNet to counter data sparseness problem. Hawkins [64] built up a WSD system that works with frequency and contextual information that is based on WordNet.

Fellbaum et al. [64] proposed a system that made use of syntactic clustering and semantic distinctions extracted from WordNet.

WordNet is mostly used to determine semantic similarity between senses. Resnik [65] computed information content of words which is a measure of the specificity of the concept that subsumes the words in the WordNet hypernym hierarchy. Agirre and Rigau [66] employ WordNet to determine the conceptual distance among concepts whereas Mihalcea and Moldovan [67] exploit semantic density and WordNet glosses in an all words word sense disambiguation. Lin [68, 69, 70] described a semantic similarity measure where similarity between two objects is defined to be the amount of information contained in the commonality between the objects divided by the amount of information in the description of the objects. We used the same measure for the work conducted in this thesis. Detailed information is given in Section 3.2.3.

Other approaches using WordNet are Jiang [71], Agirre and Martinez [18], Agirre et al. [72], Haynes [73] and Banerjee et al. [74]. A combination of MRDs and WordNet has also been tried with some success (Litkowski, 2000, 2001; Mihalcea and Moldovan, 1998) [75, 76, 77].

2.3 Information Sources for WSD

There are various information sources or feature types used in WSD regardless of the type of the approach. To disambiguate a word, various kinds of information, including syntactic tags, word frequencies, collocations, semantic context, role-related expectations, and syntactic restrictions can be considered.

In Agirre and Martinez [72], a comparison of WSD systems has been made based on the information source they used. Some of these sources are as follows:

Frequency of senses: Frequency information is used to measure the likelihood of each possible sense appearing in the text. Therefore this information is generally used in statistical approaches and it is generally learned from hand-tagged data such as SemCor corpus. Interestingly very few WSD approaches outperform the "most frequent sense" heuristic. WordNet senses are ordered according to the frequencies of the senses in the SemCor corpus.

Part of speech (POS): Part of speech tagging is regarded as the first step of the disambiguation process if the lemmas have the same orthographic forms but different syntactic classes. It is useful because it reduces the number of possible senses a word can belong to. An orthographic form may even be unambiguous in one syntactic class whereas it has more than one sense in another. For instance, in WordNet 2.0 *handle* has 5 senses as a verb, but only one sense as a noun. The impact of knowledge resources on WSD in examined in Gaustad [78]. The results show that accurate POS information is beneficial for WSD and that including the POS of the ambiguous word itself as well as POS of the context increases the disambiguation accuracy.

Morphology: It is defined as the relation between derived words and their roots. For instance, the noun *agreement* has 6 senses, its verbal root *agree* 7. A stemmer tries to reduce various forms of a word to a single stem. Since English is a language with little inflectional morphology, it is not certain that using morphology will lead to significant improvements in WSD. With other languages, such as German or Italian, morphology is of greater influence.

Collocations: Collocation is the relationship among any group of words that tend to co-occur in a predictable configuration. Disambiguation relies heavily on collocational information. For example, the noun *match* has 9 senses. However, it has only one possible sense in "*football match*". It is observed that collocations are strong indicators if they are learned from hand-tagged corpora. Although they are strong, they should be used with

other sources. They should not be treated as rules for sense-filtering alone (as in Hirst 1987 [79] or Dahlgren, McDowell, and Stabler 1989 [80]).

Semantic word associations: These can be classified as follows:

i. *Taxonomial organization:* This refers to the classification of words in a hierarchy and the lexical-semantic relationships holding between words such as a *dog* is a kind of *animal*. This kind of information can be extracted from ontologies like WordNet.

ii. *Situation and Topic:* Information about the situation or topic enables a WSD system to see the ambiguous word in a broader context. For example, if the word *mouse* is used in an office situation and the topic is computer use, the most probable sense of the word *mouse* will be "computer tool", not "animal". Semantic word associations around topic and situation are powerful when learned from hand-tagged corpora. Associations learned from MRDs can also be useful.

iii. *Argument-head relations:* These relations provide important clues for disambiguation such as the relationship between *dog* and *bite* in the sentence "the dog bit the man."

Subcategorization information: Subcategorization refers to certain kinds of relations between words and phrases. For example the verb *want* can be followed by an infinitive, as in *"I want to fly to Istanbul"*, or a noun phrase, as in *"I want a flight to Istanbul"*. But the verb find cannot be followed by an infinitive. For example *"I found to fly to Istanbul."* is not a correct sentence. Verbs have several possible patterns of arguments. A particular set of arguments that a verb can appear with is referred to a subcategorization frame. Subcategorization frames capture syntactic regularities about complements.

Agirre and Martinez [72] made a comparison between the contributions of the above resources to WSD. According to their observations, if learned from hand-tagged corpora,

collocations and semantic word associations are the most important knowledge types for WSD, but they also mentioned that syntactic cues are equally reliable. On the other hand, taxonomical information was found to be very weak.

2.4 General Approach and Related Work

As can be seen from the previous sections, most of the studies in WSD are concentrated so far on supervised learning since these algorithms usually achieve the best performance. This fact is shown by the results of SENSEVAL-2 [81]. However these systems have the drawback that they are only applicable to the words for which sense tagged data is available, and they are strongly dependent on the labeled training data available. In addition, almost all of these systems make use of the picking the first sense heuristic.

Supervised systems and first sense heuristic has come to a natural bound of approximately 70% precision [82]. In order to improve this value, benefits of knowledge-based or unsupervised alternatives should be examined.

In this thesis, we present a disambiguation algorithm that relies on the information gained from syntactic context and sense similarity. The WSD approach used in the system is unsupervised because no sense-tagged data is needed for training. All details about the system will be given in Chapter-3. In this section, a number of WSD systems that have employed syntactic context in the past are introduced.

The algorithm used in this thesis is very similar to the algorithm presented in Lin [68]. Lin [68] mentions the same problems that supervised algorithms face with: need for a large sense-tagged training set and data sparseness. In order to overcome these problems, Lin proposes an algorithm adopting the intuition that “two different words are likely to have similar meanings if they occur in identical local contexts”. This means that the same

knowledge sources are used for all words and that instead of building separate classifiers for each word, past usages of other words are used to disambiguate the current word.

In Lin's work [68], using a broad-coverage parser, a 25-million-word Wall Street Journal corpus was parsed and dependency relationships were extracted. Dependency relationships were stored as dependency triplets which were composed of the type of the dependency relation, the word related to the ambiguous word via the dependency relation and information on whether the word is the head or the modifier in the relationship. A local context database was constructed for each of these triplets (local contexts). An entry in the database consisted of twenty words found from the training corpus that were most likely to occur in that particular local context.

In the disambiguation process of Lin [68], input is parsed and local contexts are extracted. Local context database is searched to find the words that occurred in the same local context as the ambiguous word. Based on a similarity maximization algorithm between the words, the polysemous word is disambiguated. The algorithm used in the similarity maximization part of both Lin's work and our work is very similar to the algorithm presented in Resnik [65].

Since Wall Street Journal is mostly business news, Lin only used the "press reportage" of SemCor corpus for testing. An accuracy of 56.1% is reported for nouns. The performance of the baseline, picking the first sense, is reported as 58.9%.

Li, Szpakowics, and Matwin [62], presents an algorithm for automatic word sense disambiguation, based on lexical knowledge contained in WordNet and on the results of surface-syntactic analysis. The algorithm was designed to support text analysis with minimal precoded knowledge. They proposed a set of heuristic rules that are based on the idea that objects of the same or similar verbs are similar. Disambiguation is done by computing the semantic similarity between words and applying the heuristic rules based on this similarity.

Stetina et al. [83] achieved good results with syntactic relations as features compared to the frequency baseline. To evaluate the algorithm, they randomly selected 15 files from the set of 103 files of the sense tagged section of the Brown Corpus. Each tested file was removed from the set and the remaining 102 files were used for learning. They achieved an accuracy of 84.2% whereas the accuracy of the most frequent sense was 77.8%. They used a measure of semantic distance based on WordNet to find similar features. The features were extracted using a statistical parser [84] and consisted of the head and modifiers of each phrase.

Martinez et al. [85] discuss the contribution of various syntactic features to WSD: grammatical relations coded as the presence of adjuncts/arguments in isolation or as subcategorization frames, and instantiated grammatical relations between words.

Martinez extracted the set of syntactic features using Minipar [86]. Two types of syntactic relations were taken into account: instantiated grammatical relations (IGR) and grammatical relations (GR). IGR are triples of [wordsense relation value] where the value is either the word form or the lemma. For GR, bigrams [wordsense relation] and n-grams [wordsense relation1 relation2...] are stored. Three types of n-grams are defined: all subcategorization information including part of speech given by Minipar, subcategorization information occurring in the sentence, and all dependencies in the parse tree. Their results on the Senseval-2 lexical sample data for English indicate that both types of syntactic relations, IGR and GR, together provide the best F-score (IGR getting better precision than coverage and GR lower precision, but higher coverage).

Gaustad [78] also investigated whether structural syntactic information is helpful for lexical semantic disambiguation or not. He used the Alpino dependency parser for Dutch in order to extract the dependency relations. He reported that using dependency relations in conjunction with the lemma of the ambiguous word and its part of speech as features performs better than the model including only immediately left and right context.

Chapter 3

WORD SENSE DISAMBIGUATION BASED ON SENSE SIMILARITY AND SYNTACTIC CONTEXT

3.1 Introduction

Most corpus-based WSD algorithms make use of the previous usages of a polysemous word in order to disambiguate that word. These algorithms basically rely on the intuition that "two occurrences of the *same* word have *identical* meanings if they have *similar* local contexts". In order for this method to have the desired consequences; a polysemous word must be observed many times before making a good classifier. Knowing that there are nearly 15,000 polysemous nouns in WordNet 2.0 having approximately 40,000 senses the corpus must contain millions of words. If the corpus fails to contain a polysemous word or all the senses of a word, the algorithm can lead to false results.

In this thesis, we make use of a WSD algorithm that relies on the intuition that "two *different* words are likely to have *similar* meanings if they occur in *similar* local contexts". By this approach, a generalization can be made since it allows using the knowledge learned for one word on other words; instead of using a separate classifier for each polysemous word. Hence, the size of the training corpus can be much smaller. Moreover, this method is able to disambiguate infrequent or unseen words in the training corpus.

The basic steps of the algorithm are as follows:

i. Parse the training corpus and build up a local context feature database by extracting local context features according to some rules.

ii. Similarly, parse the input text and extract the local context of each ambiguous word. One local context can be composed of only one local context feature or several local context features.

iii. For each ambiguous word *w*, search the local context feature database for each feature in its local context and find words that are in a similar local context as *w*. These words are the similar words set *S* of *w* which will be used in sense selection.

iv. Select a sense of *w* that maximizes the similarity between *w* and its similar words set *S* that was found in the previous step.

Consider the sentence below:

Jane applied for a job. (3.1)

The noun *"job"* has 15 possible meanings in WordNet 2.0. To disambiguate the noun *"job"*, we consider the other words that appeared in a similar context as *"job"*. According to our system, the local context of *"job"* is consisting of only one local context feature "*N:prep-for:apply:head:V"*. "*N:prep-for:apply:head:V"* means that there is a relation between the verb "*apply"* and the noun "*job"* via the preposition *"for"*. In Figure 3.1, the words that has the same local context feature "*N:prep-for:apply:head:V"* are presented. The numbers in table indicate the likelihood ratios of the words in that local context feature. The meaning of "job" in the sentence (3.1) can be determined by choosing one of its 15 senses that is most similar to the meanings of the words in Figure 3.1.

N:prep-for:apply:head:V			
license	419.46	citizenship	100.85
membership	324.86	asylum	92.37
job	299.67	visa	84.21
approval	241.81	status	72.32
patent	233.65	grant	69.68
permission	207.59	exemption	59.04
amnesty	198.54	legalization	58.55
permit	166.21	noun.person	54.34
listing	141.75	refund	52.63
loan	141.00	residence	50.20

Figure 3.1: Words that have a relation with *"apply"* via *"for"*

Likelihood Ratio

The likelihood ratio of a local context feature and a word is obtained by treating the word and the local context feature in a tuple as a bigram and applying the log-likelihood formula presented in Dunning [87]. The log-likelihood test compares two hypotheses about a word w_1 and a local context feature w_2:

Hypothesis 1: $H_1 = P(w_2|w_1) = p = P(w_2|\ not\ w_1)$

Hypothesis 2: $H_2 = (\ P(w_2|w_1) = p_1\ ) \neq\ (p_2 = P(w_2|not\ w_1))$

Assuming binomial distribution, the log likelihood ratio is calculated as follows:

$$\log \lambda = \log \frac{L(H_1)}{L(H_2)} = \log \frac{b(c_{12}, c_1, p)b(c_2 - c_{12}, N - c_1, p)}{b(c_{12}, c_1, p_1)b(c_2 - c_{12}, N - c_1, p_{2)}}$$

$$= \log L(c_{12}, c_1, p) + \log L(c_2 - c_{12}, N - c_1, p) - \log L(c_{12}, c_1, p_1) - \log L(c_2 - c_{12}, N - c_1, p_2)$$

$$\text{where } L(k,n,x) = x^k (1-x)^{n-k}$$

Note that c_1 is the frequency of word w_1, c_2 is the frequency of the local context feature w_2, c_{12} is the frequency of bigram (w_1,w_2), that is the frequency of word w_1 appearing in the local context feature w_2, N is the number of total words in the corpus, and

$$p = \frac{c_2}{N}, p_1 = \frac{c_{12}}{c_1}, p_2 = \frac{(c_2 - c_{12})}{(N - c_1)}.$$

The higher the value, the more likely that word w_1 is seen in the local context feature w_2.

This chapter is structured as follows: First, we present the resources needed for the algorithm. In section 3.3 and section 3.4, the concepts of local context, local context feature and local context feature database are explained. Later, the disambiguation algorithm is presented in section 3.5. Finally, we conclude this chapter with the results obtained from the experiments.

3.2 Resources of the Algorithm

In order to implement this algorithm the following resources are needed:

i. an untagged training corpus,
ii. a concept hierarchy,
iii. a similarity measure,
iv. a broad-coverage parser.

In this section, all of the above resources are presented one by one.

3.2.1 Training Corpus

For the training corpus, we used a 100-million-word Wall Street Journal corpus of the periods presented below:

Source	Period	Original Publication(s)
WSJ	1987-9	ACL/DCI and CSR WSJ0
WSJ	1990-2	TIPSTER Vol. 2, version 2
WSJ	1992-4	CSR LM-1 corpus

The WSJ 1992 collection on TIPSTER covers only the first three months of the year. Data from the remaining months of 1992 were taken from the LDC, along with all of 1993 and the first three months of 1994.

3.2.2 Concept Hierarchy (WORDNET 2.0)

As the concept hierarchy WordNet 2.0 is used. Information about the general WordNet structure is given in section 2.2.3.

3.2.3 Similarity Measure

In order to find the similarity between the senses of a polysemous word and another word we need a similarity measure. There is a strong relationship between a word's semantics (or meaning) and its syntactic behavior. As discussed in Weeds [88], if two words mean similar things then they will occur in documents in similar contexts. Accordingly, we can attempt to calculate the similarity between all senses of two words by

finding all of the contexts that each appears in. If a description of a word is made up of the contexts it appears in and their associated frequencies then we can mathematically measure the similarity between these descriptions. Many measures have been proposed to calculate the similarity between the descriptions of words; we used WordNet Similarity Package v.0.15 which implements the semantic relatedness measures described by Leacock & Chodorow (1998) [63], Jiang & Conrath (1997) [71], Resnik (1995) [89], Lin (1997) [68], Hirst & St-Onge (1998) [90], Wu & Palmer (1994) [91], the adapted gloss overlap measure by Banerjee and Pedersen (2002) [50], and two measures based on context vectors by Patwardhan (2003) [92]. This package is able to calculate the similarity between either two words or two senses (or synsets). We use it to find the similarity between two senses.

We used Lin's measure [68] in the experiments conducted for this thesis.

Lin's Measure[2]

This measure is derived from the following assumptions (Lin, 1997) [68]:

i. The commonality between the words *A* and *B* is measured by *I(common(A,B))* where *common(A,B)* is a proposition that states the commonalities between *A* and *B* and *I(s)* is the amount of information contained in the proposition *s*.

ii. The differences between the words *A* and *B* is measured by

$$I(describe(A,B)) - I(common(A,B))$$

where *describe(A,B)* is a proposition that describes what *A* and *B* are.

iii. The similarity between *A* and *B*, *sim(A,B),* is a function of their commonality and differences. That is,

$$sim(A,B) = f\,(\, I(common(A,B))\,,\, I(describe(A,B)))$$

where the domain of function *f(x,y)* is $\{\, (x,y) \mid x \geq 0\,,\, y > 0\,,\, y \geq x \,\}$.

[2] This section is taken from Lin(1997) [68]

iv. Similarity is independent of the unit used in the information measure.

According to the Information Theory (Cover and Thomas, 1991), $I(s) = -\log_b P(s)$, where $P(s)$ is the probability of s, and b is the unit. When $b=2$, $I(s)$ is the number of bits needed to encode s. Since $\log_b x = \frac{\log_{b'} x}{\log_{b'} b}$, assumption *(iv)* means that the function f must satisfy the following condition:

$$\forall c > 0, f(x, y) = f(cx, xy)$$

v. Similarity is additive with respect to commonality.

If *common(A,B)* consists of two independent parts, then the *sim(A,B)* is the sum of the similarities computed when each part of the commonality is considered. In other words: $f(x_1 + x_2, y) = f(x_1,y) + f(x_2,y)$. Therefore, $\forall y, f(0, y) = f(x + 0, y) - f(x, y) = 0$, which means that when there is no commonality between A and B, their similarity is 0, no matter how different they are.

vi. The similarity between a pair of identical objects is 1.

When A and B are identical, knowing their commonalities means knowing what they are, i.e., $I(common(A,B)) = I(describe(A,B))$. Therefore, the function f must have the following property:

$$\forall x, f(x, x) = 1$$

vii. The function $f(x,y)$ is continuous.

Similarity Theorem: The similarity between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B and the information needed to fully describe what A and B are.

$$sim(A, B) = \frac{\log P(common(A, B))}{\log P(describe(A, B))}$$

Proof: To prove the theorem, we need to show $f(x,y) = \frac{x}{y}$. Since $f(x,y) = f(\frac{x}{y}, y)$ (due to assumption *(iv)*), we only need to show that when $\frac{x}{y}$ is a rational number, *f(x,y)* = $\frac{x}{y}$. The result can be generalized to all real numbers because *f* is continuous and for any real number, there are rational numbers that are infinitely close to it.

Suppose *m* and *n* are positive integers.

$$f(nx,y) = f((n-1)x,y) + f(x,y) = n\,f(x,y) \qquad \textit{(due to assumption (v))}$$

Thus,

$$f(x,y) = \frac{1}{n} f(nx,y).$$

Substituting $\frac{x}{n}$ for *x* in this equation:

$$f(x,y) = nf(\frac{x}{n}, y) = nf(x,ny) = n(\frac{1}{m} f(mx,ny))$$

Since $\frac{x}{y}$ is rational, there exist *m* and *n* such that $\frac{x}{y} = \frac{n}{m}$. Therefore,

$$f(x,y) = \frac{n}{m} f(\frac{mx}{ny},1) = \frac{n}{m} f(1,1) = \frac{n}{m} = \frac{x}{y}$$

For example, Figure 3.2 is a fragment of the WordNet. The nodes are synsets. The links represent hypernym relationships. For every synset *C, logP(C)* can be defined as the probability that a randomly selected noun refers to an instance of *C*. The probabilities are estimated by the frequency of synsets in SemCor.

If *x* is a *hill* and *y* is a *coast*, the commonality between *x* and *y* is that *"x is a geological-formation and y is a geological-formation"*. The information contained in this statement is *-2xlogP(geological-formation)*. The similarity between the synsets *hill* and *coast* is:

$$sim(hill, coast) = \frac{2 \times \log P(geo\log ical - formation)}{\log P(hill) + \log P(coast)}$$

Generally speaking,

$$sim(A, B) = \frac{2 * \log P(\bigcap_i C_i)}{\log P(A) + \log P(B)}$$

where $P(\bigcap_i C_i)$ is the probability that an object belongs to all the maximally specific super classes (C_i s) of both A and B.

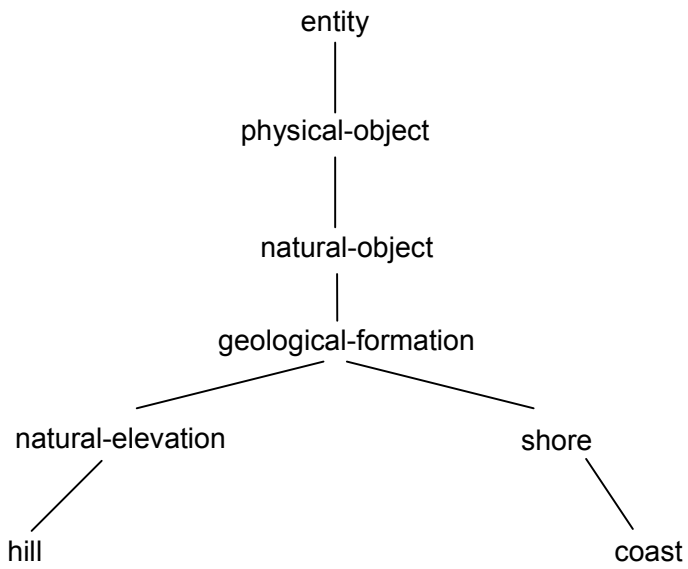


Figure 3.2: A fragment of WordNet

3.2.4 Broad-coverage Parser (MINIPAR)

Parsing means taking an input and producing a kind of structure for it. A *parser* for a natural language such as English, Turkish, etc. is a program that constructs derivations or phrase structure trees or dependencies for the sentences of that language. It supplies a correct grammatical analysis for a given sentence by separating it into its constituents and labeling each of them. It may also offer additional information, such as the semantic class (e.g., Person, Physical Object) of each word and the functional class (e.g., Subject, Direct Object) of each constituent of the sentence.

We chose to use MINIPAR [86] in order to extract the local context features from the text corpus. MINIPAR is a principle-based, broad-coverage parser for English [86]. It represents a sentence as a network of nodes and links, where the nodes represent grammatical categories and the links represent types of dependency relationships. In the Table 3.1 and Table 3.2 the lists of grammatical categories and dependency relationships are presented respectively.

MINIPAR achieves about 88% precision and 80% recall with respect to dependency relationships [68], evaluated on the SUSANNE corpus, a subset of the Brown Corpus of American English. In (Alam, 2002) [93], they also performed a test on 584 sentences. Instead of checking the parse trees, they checked the frames corresponding to the sentences. They counted all the errors caused by MINIPAR and its accuracy was found 77.6% in terms of correctly parsed sentences.

As reported in (Alam, 2002) [93], the majority of MINIPAR errors fall in the following categories:

1. Tagging errors: some nouns are mistagged as verbs.

2. Attachment errors: some prepositional phrases (PP) that should be attached to their immediate preceding nouns are attached to the verbs.

3. Missing lexical entries: some domain specific words such as *download* and their usages are not in the MINIPAR lexicon. This introduces parsing errors because such words are tagged as nouns by default.

4. Inability to handle ungrammatical sentences: in a real world application, it is unrealistic to expect the user to enter only grammatical sentences. Although MINIPAR still produces a syntactic tree for an ungrammatical sentence, the tree is ill formed and cannot be used to extract the semantic information being expressed.

GRAMMATICAL CATEGORIES	
Det	Determiners
PreDet	Pre-determiners
PostDet	Post-determiners
NUM	numbers
C	Clauses
I	Inflectional Phrases
V	Verb and Verb Phrases
N	Noun and Noun Phrases
NN	noun-noun modifiers
P	Preposition and Preposition Phrases
PpSpec	Specifiers of Preposition Phrases
A	Adjective/Adverbs
Have	have
Aux	Auxiliary verbs
Be	Different forms of “be”
COMP	Complementizer
VBE	be used as a linking verb
V_N	verbs with one argument , i.e., intransitive verbs
V_N_N	verbs with two arguments, i.e., transitive verbs
V_N_I	verbs taking small clause as complement
SentAdjunct	Sentence adjunct
U	undefined

Table 3.1: Grammatical Categories of MINIPAR

GRAMMATICAL RELATIONSHIPS	
appo	"ACME president, **--appo** → P.W. Buckman"
aux	"should ← **aux--** resign"
be	"is ← **be--** sleeping"
c	"that ← **c--** John loves Mary"
comp1	first complement
det	"the ← **det --** hat"
gen	"Jane's ← **gen --** uncle"
have	"have ← **have--** disappeared"
i	relationship btw. a C clause and its I clause
inv-aux	inverted auxiliary: "Will ← **inv-aux--** you stop it?
inv-be	inverted be: "Is ← **inv-be--** she sleeping"
inv-have	inverted have: "Have ← **inv-have--** you slept"
mod	relationship btw. a word and its adjunct modifier
pnmod	post nominal modifier
p-spec	specifier of prepositional phrases
pcomp-c	clausal complement of prepositions
pcomp-n	nominal complement of prepositions
post	post determiner
pre	pre determiner
pred	predicate of a clause
rel	relative clause
vrel	passive verb modifier of nouns
wha, whn, whp	wh-elements at C-spec positions
obj	object of verbs
obj2	second object of ditransitive verbs
subj	subject of verbs
s	surface subject

Table 3.2: Grammatical Relationships of MINIPAR

3.3 Local Context

Context is the only source of information we use to identify the meaning of a polysemous word. A great deal of work done in WSD uses local context as the primary information source. However, every system has its own definition of a local context. It is generally considered as a small window of words surrounding a target word in a text. In

these cases, the local context is generally regarded as all words or characters falling within some window without considering any features or any kinds of relations. Below are some features that are taken into consideration in local context selection in previous WSD work:

Distance: Examining a context of a few words around the target word to disambiguate is the mostly used method in WSD. In this method, there is always a question concerning the optimal window size *N*.

Yarowsky [94, 95, 96] examines different windows of local context, including 1-contexts, *k*-contexts, and words pairs at offsets -1 and -2, -1 and + 1, and +1 and +2, and sorts them using a log-likelihood ratio to find the most reliable evidence for disambiguation. He makes the observation that the optimal value of *k* varies with the kind of ambiguity: he suggests that local ambiguities need only a window of $k = 3$ or 4, while semantic or topic-based ambiguities require a larger window of 20-50 words. No single best measure is reported, suggesting that for different ambiguous words, different distance relations are more efficient.

Collocation: The term "collocation" has been used in various senses in WSD work. Yarowsky [94] adapted the definition of collocation as "the co-occurrence of two words in some defined relation." and observed a strong tendency for words to exhibit only one sense in a given collocation.

Syntactic relations: In this thesis, the local context of a word is defined in terms of syntactic dependencies between the word and the other words in the same sentence. Dependency grammars generate parses where words in a sentence are related directly to the word which is its syntactic head. As mentioned before, we used MINIPAR in order to generate a parse including dependency relationships between words for a sentence. Some examples of dependencies generated by MINIPAR can be found in Figure 3.3.

In Figure 3.3, the labels on the arrows indicate the grammatical relationships generated by MINIPAR which are shown in detail in Table 3.2. *A local context feature* of a word W_1 is defined as a tuple as follows:

$$POS_1 : relation : W_2 : position : POS_2$$

where POS_1 is the part of speech of word W_1; *relation* is the grammatical relationship between W_1 and W_2; W_2 is the word related to W_1 via the grammatical relationship; *position* is the indicator of W_2 whether it is the syntactic *head* or the syntactic *modifier(mod)* in the grammatical relation and POS_2 is the part of speech of word W_2. Part of speech tagging is also done by MINIPAR and the possible part of speech tags are presented in Table 3.1.

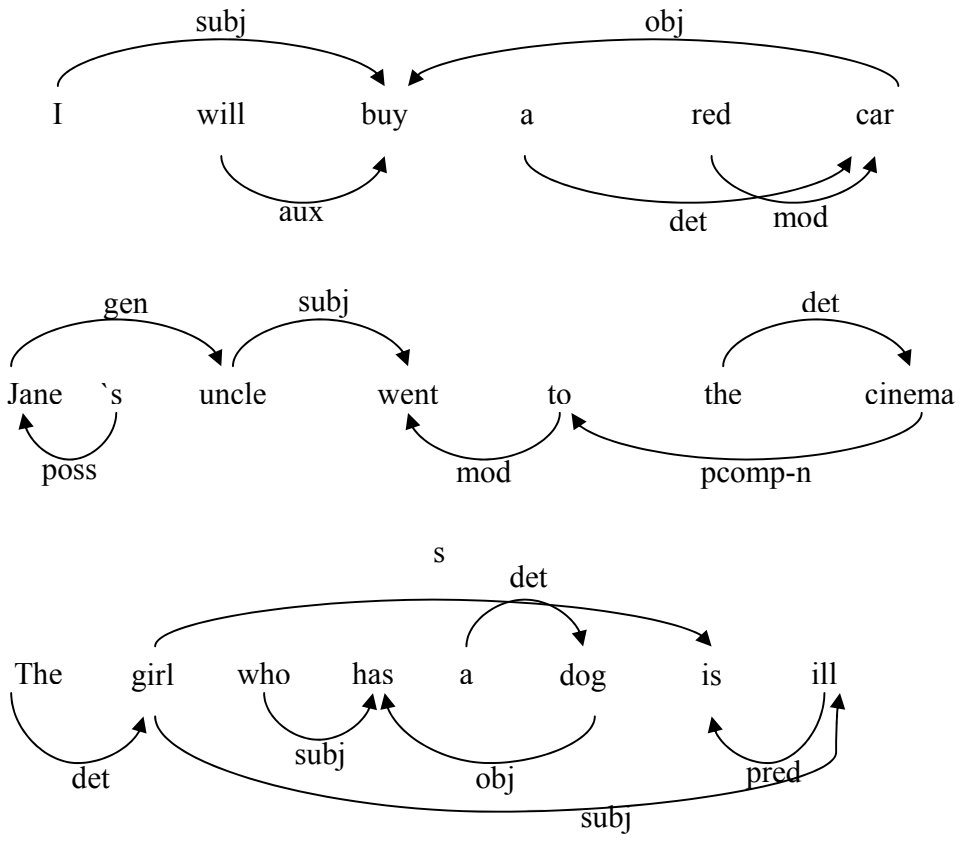


Figure 3.3: Examples of dependencies generated by MINIPAR

A ***local context*** is a collection of local context features for different W_2. Below the local context of each word for the first sentence (dependency diagram) in Figure 3.3 is presented:

word	local context
buy	[(V : aux : will : mod : Aux) (V : subj : I : mod : N) (V : obj : car : mod : N)]
will	[(Aux : aux : buy : head : V)]
I	[(N : subj : buy : head : V)]
car	[(N : obj : buy : head : V) (N : mod : red : mod : A) (N : det : a:mod :Det)]
red	[(A : mod : car : head : N)]
a	[(Det : det : car : head : N)]

3.4 Local Context Feature Database

In order to construct a local context feature database[3], initially, we have parsed the 100-million-word training corpus with MINIPAR and extracted local context features. Later, we have introduced two types of modifications: function words are removed and they are bridged with semantic relations. Below, the general rules for excluding function words from the database are presented:

1. If one of the words in the dependency relationship is an auxiliary verb, that is if POS_1 or POS_2 is equal to *"Aux"* and also the relationship is *"aux",* then we delete that dependency relationship from the database. Auxiliary verbs (will, shall, should, can, etc.) do not provide us any information about the meaning of a word.

2. If one of the words in the dependency relationship is a sentence adjunct, that is if POS_1 or POS_2 is equal to *"SentAdjunct",* then we delete that dependency relationship from the database. Sentence adjuncts are always optional because they do not depend on the

[3] The local context feature database is available at http://home.ku.edu.tr/~dyuret/students/bmutlum/thesis/code-base.htm

verb, and they may occur initially or finally. For example, "*When you receive that card, you should know that you are not being invited to join the party."*

In this case, there is also a link to the root node represented by *"fin"*. If one of the words in the dependency relationship is *"fin"*, that is if W_1 or W_2 is equal to *"fin"*, we delete that dependency relationship from the database.

3. A complementizer is a conjunction which marks a complement clause. If one of the words in the dependency relationship is a complementizer, that is if POS_1 or POS_2 is equal to *"COMP",* then we delete that dependency relationship from the database. An example for a complementizer is: *"I know that he is here."*

4. If the dependency relation is *"punc"* or if POS_1 or POS_2 is equal to *"U"* or *"have",* then these dependency relationships are deleted from the database.

5. Contractions (`m, `s, `re, etc.) are deleted from the database.

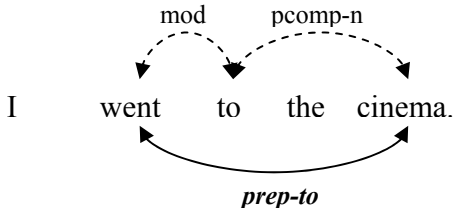
6. Some words are used very frequently in English. One of these words is *"one"*. Therefore, since the information gained from these kinds of words does not contribute to the disambiguation process, dependency relationships including these words[4] are deleted from the database.

7. Determiners are deleted from the database, i.e. a, an, the.

[4] The list of these words are available at http://home.ku.edu.tr/~dyuret/students/bmutlum/thesis/code-base.htm

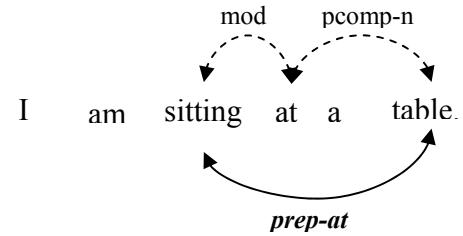
In addition to the dependency relationships produced by the parser, we created new dependency relationships by bridging function words. The rules about how to create these will be presented on example sentences. Consider the five sentences and related parser outputs below :

(1)



go	V:mod:to:mod:Prep
to	Prep:mod:go:head:V
to	Prep:pcomp-n:cinema:mod:N
cinema	N:pcomp-n:to:head:Prep

(2)



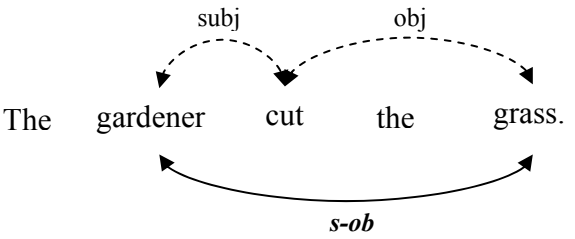
sit	V:mod:at:mod:Prep
at	Prep:mod:sit:head:V
at	Prep:pcomp-n:table:mod:N
table	N:pcomp-n:at:head:Prep

Rule-1:

In the sentences (1) and (2), consider the relations between the words *go-to-cinema* and *sit-at-table*. The parser produces the relationship between a word and its adjunct modifier for word pairs *go-to* and *sit-at* where the prepositions "to" and "at" are the adjunct modifiers for the verbs "go" and "sit"; and a nominal complement of preposition relations for *to-cinema* and *at-table* word pairs where "cinema" and "table" are nominal complements for prepositions "to" and "at". But it does not produce any relations for *go-cinema* and *sit-table* word pairs since there is no syntactic dependency in these pairs. However there is a semantic dependency in each of these pairs. So, as a rule, we introduce new relations for every nominal complement of prepositions, and the relation is different for each preposition such as *prep-to*, *prep-at*, *prep-with*, etc. In this case below tuples are produced for sentence (1) and sentence (2) respectively:

For (1):	go	V:prep-to:cinema:modifier:N
	cinema	N:prep-to:go:head:V
For (2):	sit	V:prep-at:table:modifier:N
	table	N:prep-at:sit:head:V

(3)



cut	V:subj:gardener:mod:N
gardener	N:subj:cut:head:V
cut	V:obj:grass:mod:N

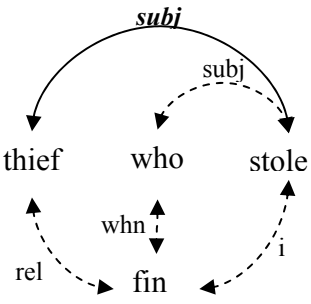
grass	N:obj:cut:head:V

Rule-2:

For the sentence (3), the parser produces a subject relation for *gardener-cut* pair where "gardener" is the subject and an object relation for *cut-grass* pair where "grass" is the object. There is no syntactic relation in the *gardener-grass* pair. For these kind of subject and object pairs, as a rule, we introduce the new "s-ob" relation resulting in tuples:

gardener	N:s-ob:grass:modifier:N
grass	N:s-ob:gardener:head:N

(4) The thief who stole her bag is caught.



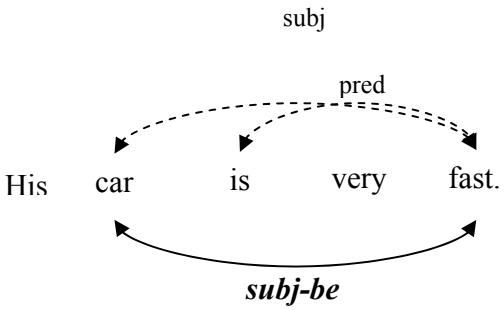
thief	N:rel:fin:mod:C
fin	C:rel:thief:head:N
fin	C:whn:who:mod:N
who	N:whn:fin:head:C
fin	C:i:steal:mod:V
steal	V:i:fin:head:C
steal	V:subj:who:mod:N
who	N:subj:steal:head:V

Rule-3:

In the sentence (4), there exists a relative pronoun "who", which is the syntactic subject of the verb "steal". But, this relative pronoun does not give us all the semantic information. So, as a rule, for the subordinate clauses constructed from relative pronouns (who, which, that), we assign the main clause's subject to be the subject of the subordinate clause's verb, like in the example pair *thief-steal*:

steal	V:subj:thief:modifier:N
thief	N:subj:steal:head:V

(5)



be	VBE:pred:fast:mod:A
fast	A:pred:be:head:VBE
fast	A:subj:car:mod:N
car	N:subj:fast:head:A

Rule-4:

From the sentence (5), we get a subject relation for the pair *car-fast* where "car" is the subject. This is syntactically correct but it would be better to give a different name to the

subject relation that is formed by the verb “be”. Therefore, a new relation is introduced with the name “subj-be” for such cases:

fast	A:subj-be:car:modifier:N
car	N:subj-be:fast:head:A

Different from Lin [68], rules that are explained above are applied to the output of the parser to obtain the new dependency tuples for the local context feature database. An entry in the local context feature database is composed of one local context feature and a set of 20 words which have top-20 likelihood ratios with that local context feature. In Figure 3.4, different database entries are presented for different local context features.

For the local context feature *“N:subj:employ:head:V”,* the subjects of employ that have the highest 20 likelihood ratios are presented. So, *“plant”* is the most probable noun that is supposed to occur as the subject of the verb *“employ”* according to our local context feature database. The most probable verb to occur in a subject relationship with the noun “*student*” is “*learn*”. We can infer from the database that the thing that is most applied for is *“license”* and the noun *“girl”* is mostly modified by a quantifier.

There are some expressions as “*noun.quantity*” or “*noun.person*” in the local context feature database. These expressions are obtained by exposing the output of the parser to a named entity recognition process. Lin [68], also performed named entity recognition on the output of the parser. *“noun.person”* includes all proper names recognized as a person, *“noun.group”* includes all proper names recognized as organization or corporation names, *“noun.location”* are the proper names used for addresses and geological forms, *“noun.time”* is the proper name for dates, *“noun.possession*” is the proper name regarding money and *“noun.quantity”* is used for numbers.

N:subj:employ:head:V			
plant	947.30	noun.person	51.79
company	380.15	venture	41.58
unit	148.10	industry	41.37
facility	129.88	kitchen	40.80
operation	126.82	noun.possession	27.45
factory	125.74	enterprise	26.42
division	121.38	manufacturer	26.20
firm	81.04	he	23.67
business	73.01	arrowhead	19.91
mill	70.60	digital	19.65

V:subj:student:mod:N			
learn	442.88	write	123.75
attend	374.50	receive	117.87
take	285.13	know	104.90
study	185.36	choose	101.60
get	184.58	protest	94.61
graduate	151.54	enter	89.29
read	144.28	enroll	78.05
have	140.48	apply	77.46
use	137.84	perform	77.41
work	125.44	want	76.17

N:prep-for:apply:head:V			
license	419.46	citizenship	100.85
membership	324.86	asylum	92.37
job	299.67	visa	84.21
approval	241.81	status	72.32
patent	233.65	grant	69.68
permission	207.59	exemption	59.04
amnesty	198.54	legalization	58.55
permit	166.21	noun.person	54.34
listing	141.75	refund	52.63
loan	141.00	residence	50.20

A:mod:girl:head:N			
noun.quantity	481.79	adolescent	38.53
little	278.97	black	34.77
pretty	131.91	rich	30.48
beautiful	103.80	other	28.94
nice	74.59	twin	25.14
bad	64.17	dancing	24.82
young	49.81	slave	22.36
poor	45.45	small	21.76
local	41.28	suburban	20.66
good	39.02	unmarried	20.30

Figure 3.4: Examples of local context feature database entries

3.5 Disambiguation Algorithm

Having explained the concept of local context, local context feature and local context feature database, the steps of the algorithm is presented in this section in detail.

Step-1: Input text is parsed and local context features of each word are extracted. Let LC_w denote the set of local context features of word w. (The set of local context features make up the local context of a word.)

For each word ambiguous word w in the input text:

Step-2: Search the local context feature database for every local context feature *lc* in LC_w in order to find the words that appeared in a similar local context as *w*. Let these words be denoted as Selectors_w :

$$Selectors_w = (\bigcup_{lc \in LC_w} C(lc)) - \{w\}$$

where *lc* is a local context feature and *C(lc)* is a set of (word likelihood) pairs as shown in Figure-8.

Step-3: Select a sense *s* of *w* that maximizes the similarity between *w* and $Selectors_w$.

For the similarity maximization process, suppose that $w = W_0$,
selectors of W_0 are $Selectors_w = \{W_1, W_2, \ldots . W_k\}$, and
the word W_i has n_i senses which are denoted as $\{s_{i1}, s_{i2}, \ldots, s_{in}\}$.

Step-3.1: Construct a similarity matrix which is divided into (k+1)×(k+1) blocks as shown in Figure 3.5. (One block for each word $\{W_0 , W_1 , W_2, \ldots . W_k\}$.)
This matrix is used to keep the similarity values between every sense of each word. For example, suppose that the word W_2 has 4 senses and the word W_7 has 8 senses. Then the block B_{27} will be a 4×8 matrix consisting of the similarity values between every sense combination of W_2 and W_7.
The blocks on the diagonals will be all consisting of 0s. The actual similarity values on the diagonals are 1 since the similarity value between the same senses of a word is 1. However, we do not want these values to be effective in our similarity maximization process, so we assign them as 0. Similarity values lower than 0.2 are considered as noise and are ignored. The maximum similarity value between two senses is 1 and the minimum similarity value is 0 according to Lin's measure. 0.2 is selected as a threshold.

$$B_{ij}(l,m) = \begin{cases} sim(s_{il}, s_{jm}) & \text{if } i \neq j \text{ and } sim(s_{il}, s_{jm}) \geq 0.2 \\ 0 & \text{otherwise} \end{cases}$$

where $sim(s_{il}, s_{jm})$ is the similarity value found by the WordNet Similarity Package using Lin's measure.

Step-3.2: Let A be the set of polysemous words in $\{W_0, W_1, W_2, \ldots.. W_k\}$:

$$A = \{ W_i \mid n_i > 1 , 0 < i \leq k\}$$

Step-3.3: Find a sense of one of the words in A that gets the highest total support from other words. That is, find a sense which is most similar to every word. Call this sense $s_{i_{\max} l_{\max}}$:

$$s_{i_{\max} l_{\max}} = \arg\max_{s_{il}} \sum_{j=0}^{k} \sup\, port(s_{il}, W_j) \qquad 0<i\leq k$$

where s_{il} is one of the senses of word W_i (l *th* sense of W_i where $l \in [1, n_i]$) that is in the set A ($W_i \in A$) and $support(s_{il}, W_j)$ is defined as the support s_{il} gets from W_j:

$$\sup\, port(s_{il}, W_j) = \max_{m \in [1, n_j]} B_{ij}(l,m)$$

Step-3.4: The word $W_{i_{\max}}$ is disambiguated and its sense is selected as $s_{i_{\max} l_{\max}}$. $W_{i_{\max}}$ is removed from the set of ambiguous words set A.

$$A \longleftarrow A - \{W_{i_{\max}}\}$$

Step-3.5: Since $W_{i_{\max}}$ is disambiguated, the similarity values between other senses of $W_{i_{\max}}$ and senses of other words are removed.

For all l, j, m, such that $l \in [1, n_{i_{\max}}]$ and $l \neq l_{\max}$ and $j \neq i_{\max}$ and $m \in [1, n_j]$:

$$B_{i_{\max} j}(l,m) \longleftarrow 0$$

Step-3.6: If the disambiguated word is W_0 , that is $i_{max} = 0$, stop. Otherwise repeat from Step3.3.

Step-4: The sense found for the word *w* in Step-3 is assigned to the word.

	senses of W_0 s_{01} . . . s_{0n_0}	W_1 , …, W_{k-1}	senses of W_k s_{k1} . . . s_{kn_k}
s_{01} . . s_{0n_0}	0		B_{0k}
s_{11} . . s_{1n_1}	B_{10}		B_{1k}
. . .			
s_{k1} . . s_{kn_k}	B_{k0}		0

Figure 3.5: Similarity matrix

The similarity maximization algorithm used in this thesis is the same as the similarity maximization algorithm used in Lin [68].

A Walk through Example

Let's consider the ambiguous word *"work"* in the sentence below which is taken from SENSEVAL-2:

> "….the less gifted get the necessary grounding for further study or for entering the modern world of *work*."

The noun *"work"* has one local context feature in its local context when it is parsed: *"pcomp-n"* relationship with the preposition *"of"*. And preposition *"of"* has a *"mod"* relationship with the noun *"world"*. Therefore, we apply the **rule-1** and obtain the *"prep-of"* relationship between the noun "work" and the noun *"world"*. Figure 3.6 lists the words that appeared in the local context feature *"N:prep-of:world:head:N"*.

N:prep-of:world:head:N			
wonder	591.14	medicine	22.28
politics	142.60	computer	20.48
finance	65.33	fashion	20.13
banking	50.27	telecommunication	19.85
work	*41.56*	music	18.41
business	38.41	channel	18.19
difference	31.38	retailing	17.14
art	29.72	bloc	17.02
journalism	29.15	advertising	16.87
sport	24.63	computing	16.57

Figure 3.6: Words that have a relationship with *"world"* via *"of"*

The noun *"work"* has 7 senses in Wordnet 2.0 which are shown in Figure 3.7. *"work"* is used in its 3rd sense in the sentence above. We have chosen this example because first sense heuristic does not work for this case.

The noun work has 7 senses (first 7 from tagged texts)

1. (435) work -- (activity directed toward making or doing something; "she checked several points needing further work")
2. (359) work, piece of work -- (a product produced or accomplished through the effort or activity or agency of a person or thing; "it is not regarded as one of his more memorable works"; "the symphony was hailed as an ingenious work"; "he was indebted to the pioneering work of John Dewey"; "the work of an active imagination"; "erosion is the work of wind or water over time")
3. (96) employment, work -- (the occupation for which you are paid; "he is looking for employment"; "a lot of people are out of work")
4. (44) study, work -- (applying the mind to learning and understanding a subject (especially by reading); "mastering a second language requires a lot of work"; "no schools offer graduate study in interior design")
5. (41) oeuvre, work, body of work -- (the total output of a writer or artist (or a substantial part of it); "he studied the entire Wagnerian oeuvre"; "Picasso's work can be divided into periods")
6. (18) workplace, work -- (a place where work is done; "he arrived at work early today")
7. (14) work -- ((physics) a manifestation of energy; the transfer of energy from one physical system to another expressed as the product of a force and the distance through which it moves a body in the direction of that force; "work equals force times distance")

Figure 3.7: WordNet 2.0 entry for the noun *"work"*

When the hypernyms of each sense are investigated from Figure 3.8, it is seen that Sense-1 and Sense-3 are subclasses of "*act, human action, human activity"*, different from Sense-1, Sense-3 is also a subclass of "*occupation, business, job, line of work, line*". Sense-2 and Sense-5 are subclasses of "*artifact"*, Sense-4 is a subclass of "*basic cognitive process"*, Sense-6 is a kind of *"location"* and Sense-7 is a "*physical phenomenon"*. Majority of the selectors in the Figure 3.6 have "*act, human action, human activity"* senses: i.e. *politics, business, journalism, sport, medicine, banking, art, fashion, music, channel, retailing, computing,* and *advertising.* Therefore, Sense-1 and Sense-3 gets more support than other senses. Moreover, from the selector list *politics, business, journalism, sport,* and

```
Sense 1
work
      => activity
         => act, human action, human activity
Sense 2
work, piece of work
      => product, production
         => creation
            => artifact, artefact
               => object, physical object
                  => entity
               => whole, whole thing, unit
                  => object, physical object
                     => entity
Sense 3
employment, work
      => occupation, business, job, line of work, line
         => activity
            => act, human action, human activity
Sense 4
study, work
      => learning, acquisition
         => basic cognitive process
            => process, cognitive process, mental process, operation, cognitive operation
               => cognition, knowledge, noesis
                  => psychological feature
Sense 5
oeuvre, work, body of work
      => end product, output
         => product, production
            => creation
               => artifact, artefact
                  => object, physical object
                     => entity
                  => whole, whole thing, unit
                     => object, physical object
                        => entity
Sense 6
workplace, work
      => geographic point, geographical point
         => point
            => location
               => entity
Sense 7
work
      => energy
         => physical phenomenon
            => natural phenomenon
               => phenomenon
```

Figure 3.8: Hypernyms of the noun *"work"*

medicine has the sense "*occupation, business, job, line of work, line"*. Therefore, Sense-3 is selected by our system. The other senses get support only from one or two selectors.

3.6 Evaluation and Results

In order to see how well our system performs, we used the SENSEVAL-2 [81] and SENSEVAL-3 English all-words task data. Both of these data sets consist of three texts from the Penn Treebank corpus.

We only worked with nouns. Because of the similarity measure, it is impossible to do disambiguation on adverbs and adjectives. Hypernym relationship is not defined for adjectives and adverbs in WordNet. For the verbs, since they have more senses in WordNet and the difference between the senses is very subtle, the results were not promising.

There are 1136 annotated nouns in SENSEVAL-2 and 951 annotated nouns in SENSEVAL-3. Table 3.3 shows precision and recall figures for the results that are obtained by our system.

Data	Precision	Recall
SENSEVAL-2	59.11 %	40.80 %
SENSEVAL-3	56.80 %	40.00 %

Table 3.3: Results of Senseval data

There is a variety of reasons for the recall of our system to be low. The main reason is that we have eliminated the cases for which the parser produced wrong part of speech tagging. In order to find tagging errors, parser's tagging was compared with the POS tags provided with SENSEVAL data. Secondly, local context features of the nouns sometimes did not exist in the local context feature database. Thirdly, during the similarity

maximization algorithm there were times when all of the support values became zero, so no sense could be selected and the algorithm stopped.

The performance of the baseline, picking the first sense, for the SENSEVAL-2 all words task is 62.7 %. It is 60.9 % for the SENSEVAL-3 all words task.

In order to make a comparison, the top-10 performing systems in SENSEVAL-2 English all words task are presented in Table 3.4. As seen, most of the systems use sense-tagged data for training. When the systems that have the same precision and recall values are observed, it is found that they make use of the first sense heuristic for the cases their systems are unable to produce an answer. If we also use the first sense heuristic for the unanswered cases, we get 62.0% as a precision and recall value from our system.

System Name	Precision	Recall	Supervised
100 SMUaw	0.698	0.698	Yes
100 CNTS-Antwerp	0.645	0.645	Yes
100 Sinequa-LIA – HMM	0.626	0.626	Yes
98.908 UNED - AW-U2	0.583	0.577	No
98.908UNED - AW-U	0.565	0.559	No
95.552 UCLA - gchao2	0.485	0.463	Yes
95.552 UCLA - gchao3	0.482	0.461	Yes
108.5 CL Research – DIMAP	0.423	0.46	No
100 CL Research - DIMAP (R)	0.46	0.46	No
89.729 UCLA – gchao	0.508	0.456	Yes

Table 3.4: Top-10 performing systems in SENSEVAL-2

Lin [68] reported an accuracy of 56.1% for nouns. However, in the evaluation phase, he made a restriction on the test cases. Lin only used “press reportage” part of SemCor corpus

since the training corpus were mostly business news. Our results can be considered as an improvement over Lin [68] because we did not use any restrictions in the evaluation phase.

In order to make a comparison with Lin's similarity measure, we have also used the ***adapted lesk*** measure [50]. We chose the *lesk* similarity measure since it is reported to perform best among the other similarity measures in WordNet Similarity Package in a word sense disambiguation process presented in McCarthy et al. [40]. Lesk similarity measure works by finding overlaps in the glosses of the two synsets. The similarity value is the sum of the squares of the overlap lengths. A single word overlap results in a score of 1 whereas a two-consecutive-words overlap results in a score of 4.

For comparing the similarity measures **lin** and **lesk,** we conducted an experiment on a subset of SENSEVAL-2 English all-words data consisting of only *polysemous* nouns and obtained the results presented on Table 3.5. We didn't use this similarity measure in all our experiments since it requires a lot of computational time.

Measure	Precision	Recall
Lin	43.9 %	43.9 %
Lesk	51.6 %	51.6 %

Table 3.5: Comparison of similarity measures

Chapter 4

CONCLUSION AND FUTURE WORK

In the first part of this thesis an overview of WSD is presented regarding different approaches as well as different knowledge sources. Most of the work done in the past 25 years is summarized. In spite of all the work done on WSD, relative to other fields, little progress seems to have been made. WSD is still an open problem waiting to be solved. Starting from 1980's with the availability of more resources and statistical methods like Naïve Bayes, there has been an improvement on the earlier results; however, it appears that the limit of these supervised methods has been reached currently. The problems of scarce sense-tagged data and data sparseness still remain to be a barrier in front of WSD.

Another problem of WSD is the difficulty of determining or even defining word senses. Lately, WordNet has become a de facto standard for lexical resources however it also has lots of limitations. Its sense distinctions are very fine-grained and there are gaps in the WordNet hierarchy regarding the lexical relations.

Considering these, in the second part of the thesis, we presented an algorithm which did not need a sense-tagged corpus. We obtained information from syntactic local contexts and sense similarity. With the help of a principle-based broad coverage parser, we extracted the local context for each word. Different from Lin [68], we removed the function words and we constructed new semantic relationships by bridging the words connected by function words. We computed the similarity values between the words that occurred in a similar local context using a similarity metric proposed by Lin [68]. We used WordNet in order to

evaluate our system however because of the fine-grained senses and some weaknesses in the algorithm, we were able to achieve a precision about 59%. Lin [68] also presented a similar system and reported 56.1% precision with WordNet senses, that is the same evaluation system with our system. He also evaluated his system with relaxed correctness criterion. He reported nearly 70% accuracy with the relaxed evaluation criteria which is successful for an unsupervised system.

One of the deficiencies of our algorithm is that all the local context features of an ambiguous word are treated equally in the disambiguation process. However, some local context features provide a lot of information while some others provide less. We eliminated some of these local context features and introduced new ones but a statistical model can be implemented on the disambiguation algorithm. A measure of the importance of the local context features can be introduced to rank the local context features. A candidate for this measure can be the *tf-idf* weight (term frequency - inverse document frequency) which is often used for Information Retrieval. It is a statistical technique used to evaluate how important a word is to a document. This measure can be adapted to our system for ranking the importance of local context features.

Another weakness of our algorithm is the idiomatic usages of words. The intuition that similar words occur in similar contexts does not always hold. Consider the following example:

Money talks.

Suppose we want to disambiguate *"money"* which has 3 senses. The most frequent subjects of *"talk"* according to our local context feature database are, noun.person, people, peace, man, leader, executive, noun.group, president., etc. None of these words is similar to the *"wealth"* meaning of *"money"* so we fail to disambiguate such idiomatic cases. In our system, we are able to observe that *money* and *talk* are frequently used together but since

we do not have any prior knowledge about which sense of *money* is used with *talk*, we fail to disambiguate this word. Therefore, information gained from sense-tagged data should also be used in the system when the similarity value between the senses of the ambiguous word and the senses of its selectors is below a threshold.

However, in order to get the desired information from sense-tagged data, the effect of data sparseness should be reduced. Smoothing techniques such as singular value decomposition should be examined. Moreover, to overcome the need for large sense-tagged corpora, new bootstrapping methods should be introduced. Recently there has been work on bootstrapping techniques like multiple sequence alignment which has been commonly used in bioinformatics. More work should be done on these new topics in order to get extensive knowledge and improvement.

According to our observations, the future direction in WSD research should focus on unsupervised methods. Going on the other way, trying the methods that have been examined in the history of WSD, would cause a cycle in the WSD research.

BIBLIOGRAPHY

[1] Kilgarriff, A., Senseval: An exercise in evaluating word sense disambiguation programs. In Proceedings of the First International Conference on Language Resources and Evaluation (LREC 1998), 581-588, Granada, (1998).

[2] Kilgarriff, A. and Rosenzweig, J., Framework and results for English Senseval. Computers and the Humanities, 34(1-2):15-48, (2000).

[3] Kilgarriff, A., English lexical sample task description. In Proceedings of Senseval-2, Second International Workshop on Evaluating Word Sense Disambiguation Systems, 17-20, Toulouse, (2001).

[4] Carpuat, M., and Wu, D., Word Sense Disambiguation vs. Statistical Machine Translation. In Proceedings of the 43rd Annual Meeting of the ACL, 387–394, Ann Arbor, (2005).

[5] Carpuat, M., Su, W., and Wu, D., Augmenting ensemble classification for word sense disambiguation with a Kernel PCA model. In Proceedings of Senseval-3, Third International Workshop on Evaluating Word Sense Disambiguation Systems, Barcelona, (2004).

[6] Stevenson, M., Word Sense Disambiguation: The case for Combinations of Knowledge Sources, (2003).

[7] Duda, R. and Hart, P., Pattern Classification and Scene Analysis, John Wiley and Sons, New York, (1973).

[8] Rivest, Ronald L., Learning Decision Lists, Machine Learning, 2(3) (1987), 229-246.

[9] Quinlan, J.R., Induction of Decision Trees, Machine Learning, 1, (1986), 81-106.

[10] Rumelhart, D.E., Hinton, G.E., and Williams, R.J., Learning Internal Representations by Error Propagation. In Parallel Distributed Processing, Vol. 1, (1986), 318-362.

[11] Mitchell, Tom, Machine Learning, McGraw Hill, 1997.

[12] Pedersen, T., A simple approach to building ensembles of naïve bayesian classifiers for word sense disambiguation. In Proceedings of the First Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2000), 63-69, Seattle, 2000.

[13] Gale, B., Church, K., and Yarowsky, D., A method for disambiguating word senses in a large corpus. Computers and the Humanities, 1992, 26:415-439.

[14] Gale, B., Church, K., and Yarowsky, One sense per discourse. In Proceedings of the ARPA Workshop on Speech and Natural Language Processing, 1992, 233-237.

[15] Mooney, R., Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-96), University of Pennsylvania, 1996, 82-91.

[16] Florian, R., Cucerzan, S., Schafer, C. and Yarowsky, D., Combining classifiers for word sense disambiguation. Natural Language Engineering, Special Issue on Word Sense Disambiguation Systems, 2002, 8(4):327-341.

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

[18] Agirre, E., and Martinez, D., Exploring automatic word sense disambiguation with decision lists and the web. In Proceedings of the Coling 2000 Workshop:"Semantic Annotation and Intelligent Annotation", Centre Universitaire, Luxembourg, 2000.

[19] Towell, G. and, Voorhees, E., Disambiguating Highly Ambiguous Text**,** Computational Linguistics*,* 24(1), 1998, 125-145.

[20] Black, E., An Experiment in Computational Discrimination of English Word Senses, IBM Journal of Research and Development, 1988, 32:185-194.

[21] Pedersen, T., Evaluating the effectiveness of ensembles of decision trees in disambiguating Senseval lexical samples. In Proceedings of the ACL 2002 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions, Philadelphia, 2002.

[22] Ng, H. T. and Lee, H. B., Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In 34th Annual Meeting of the Association for Computational Linguistics (ACL 1996), 40-47, Santa Cruz, 1996.

[23] Brown, Peter F., Stephen, D.P., Vincent, J.D.P., and Robert, L.M., Word sense disambiguation using statistical methods. In Proceedings of the 29th Annual Meeting, Berkeley, 1991a, 264-270.

[24] Landes, S., Leacock, C., and Tengi, R., Building a semantic concordance of English, In C. Fellbaum, ed., WordNet: An electronic lexical database and some applications. Cambridge, MA.: MIT Press, 1998.

[25] Hearst, M., Noun homograph disambiguation using local context in large text corpora. In Proceedings ot the 7th Annual Conference of the UW Centre for the New OED and Text Research: Using Corpora, Oxford, 1991.

[26] Schütze, H., Dimensions of Meaning. In Proceedings of Supercomputing, 1992, 787-796.

[27] Schütze, H., Word Space. Advances in Neural Information Processing Systems 5, 1993, 895-902.

[28] Yarowsky, D., Unsupervised word sense disambiguation rivaling supervised methods. In 33th Annual Meeting of the Association for Computational Linguistics (ACL 1995), Cambridge, 1995, 189-196.

[29] Mihalcea, R. and Moldovan, D., A highly accurate bootstrapping algorithm for word sense disambiguation. International Journal on Artificial Intelligence Tools, 2001a, 10(1-2):5-21.

[30] Mihalcea, R., Bootstrapping Large Sense Tagged Corpora, in Proceedings of the 3rd International Conference on Languages Resources and Evaluations (LREC 2002), Las Palmas, Spain, May 2002.

[31] Brown, P., Lai, J. C., and Mercer, R., Aligning Sentences in Parallel Corpora. In *Proceedings of ACL-91*, Berkeley CA, 1991b.

[32] Dagan, I. and Itai, A., Word sense disambiguation using a second language monolingual corpus. Computational Linguistics, 1994, 20(4):563-596.

[33] Ide, N., Erjavec, T., and Tufis, D., Sense discrimination with parallel corpora. In Proceedings of the ACL 2002 Workshop on Word Sense Disambiguation: Recent Successes and Future Directions, 2002, 56-60.

[34] Ng, H. T., Wang, B., and Chan, Y. S., Exploiting parallel texts for word sense disambiguation. In 41th Annual Meeting of the Association for Computational Linguistics (ACL 2003), 2003, 455-462.

[35] Witten, I.H., and Bell, T.C., The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression. IEEE Transactions on Information Theory, 1991, 37(4), 1085-1094.

[36] Good, I., The populations of frequencies of species and the estimation of population parameters, Biometrika, 1953, 40:237-264.

[37] Gale, W., Good Turing smoothing without tears. Technical report, AT&T Bell Laboratories, Murray Hill, NJ, 1994.

[38] Yarowsky, D., Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. Proceedings of 14th International Conference on Computational Linguistics, COLING-92.Nantes, 454-460, 1992.

[39] Chapman, R., Roget’s International Thesaurus (Fourth Edition). Harper and Row, New York, 1977.

[40] McCarthy, D., Koeling, R., Weeds, J. and Carroll, J., Finding predominant senses in untagged text. In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, Barcelona, Spain, 2004.

[41] Schütze, H. and Pedersen, J., Information retrieval based on word senses. In Fourth Annual Symposium on Document Analysis and Information Retrieval, 1995, 161-175.

[42] Pedersen, T. and Bruce, R., Distinguishing word senses in untagged text. In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing, 1997, 197–207.

[43] Schütze, H., Automatic word sense discrimination. Computational Linguistics, 1998, 24(1):97–123.

[44] Purandare, A. and Pedersen, T., Word Sense Discrimination by Clustering Contexts in Vector and Similarity Spaces. In the Proceedings of the Conference on Computational Natural Language Learning (CoNLL), 2004.

[45] Pedersen, T., and Bruce, R., Knowledge lean word sense disambiguation. In Proceedings of the Fifteenth National Conference on Artificial Intelligence, 1998, 800–805.

[46] Lesk, M., Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In Proceedings of ACM SIGDOC Conference, 1986, 24-26.

[47] Cowie, J., Guthrie, J., and Guthrie, L., Lexical disambiguation using simulated annealing. In Proceedings of the 15th [sic] International Conference on Computational Linguistics (Coling 1992), 1992, 359-365.

[48] Sampson, G., A stochastic approach to parsing. In Proceedings of the 11th International Conference on Computational Linguistics (COLING-86), 1986, 151-155.

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

[50] Pedersen, T. and Banerjee, S., An adapted lesk algorithm for word sense disambiguation using WordNet. In Gelbukh, A., editor, Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-02), 2002.

[51] Miller, G., Leacock, C., Tengi, R., Bunker, R., and Miller, K., Five papers on WordNet. Special Issue of International Journal of Lexicography, 1990, 3(4).

[52] Véronis, J., and Ide, N., An assessment of information automatically extracted from machine readable dictionaries. In Proceedings of the fifth conference of the European chapter of the Association for Computational Linguistics, 1991, 227-232.

[53] Ide, N., and Véronis, J., Refining taxonomies extracted from machine readable dictionaries. In Hockey, Susan and Ide, Nancy (Eds.) Research in Humanities Computing II, Oxford University Press, 1993a, 145-59.

[54] Ide, N., and Véronis, J., Knowledge extraction from machine readable dictionaries: an evaluation. Third International EAMT Workshop "Machine Translation and the Lexicon", Heidelberg (Germany), 1993b.

[55] Ide, N. and Véronis, J., Introduction to the special issue on word sense disambiguation: The state of the art. Computational Linguistics, 1998, 24(1):1-40.

[56] Walker, D. E., Knowledge resource tools for accessing large text files. In Sergei Nirenburg (ed.), Machine Translation: Theoretical and methodological issues, 254. Cambridge University Press, 1987.

[57] Lenat, Douglas B. and Guha, Ramanathan V., Building large knowledge-based systems. Addison-Wesley, Reading, Massachusetts, 1990.

[58] Briscoe, Edward J., Lexical issues in natural language processing. In Klein, Ewan H. and Veltman, Frank (Eds.). Natural Language and Speech. [Proceedings of the Symposium on Natural Language and Speech, 26-27 November 1991, Brussels, Belgium.] Springer-Verlag, Berlin, 39-68, 1991.

[59] Grishman, R., MacLeod, C., and Meyers, A., COMLEX syntax: Building a computational lexicon. Proceedings of the 15th International Conference on Computational Linguistics, COLING'94, 5-9 August 1994, Kyoto, Japan, 68-272, 1994.

[60] Voorhes, Ellen M., Using WordNet to disambiguate word senses for text retrieval. Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 27 June-1 July 1993, Pittsburgh, Pennsylvania, 171-180, 1993.

[61] Richardson, R., and Smeaton, Alan F., Automatic word sense disambiguation in a KBIR application. Working paper CA-0595, School of Computer Applications, Dublin City University, Dublin, Ireland, 1994.

[62] Li, X., Szpakowics, S., and Matwin, S., A WordNet-based algorithm for word sense disambiguation. In Proceedings of the 14th International Joint conference on Artificial Intelligence, 1368-1374, Montreal, 1995.

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

[64] Hawkins, P., DURHAM: A Word Sense Disambiguation System. PhD thesis, Laboratory for Natural Language Engineering, Department of Computer Science, University of Durham, Durham, 1999.

[64] Fellbaum, C., Palmer, M., Trang Dang, H., Delfs, L., Wolff, S., Manual and Automatic Semantic Annotation with WordNet. In “Proceedings of the NAACL Workshop on WordNet and Other Lexical Resources: Applications, Customizations”. Carnegie Mellon University, Pittsburg, PA, 2001.

[65] Resnik, P., Disambiguating Noun Groupings with Respect to WordNet Senses. Proceedings of the Third Workshop on Very Large Corpora, Cambridge, Massachusetts, 54-68, 1995.

[66] Agirre, E. and Rigau, G., Word sense disambiguation using conceptual density. In Proceedings of the 16th International Conference on Computational Linguistics (Coling 1996), 16-22, 1996.

[67] Mihalcea, R. and Moldovan, D., A method for word sense disambiguation of unrestricted text. In 37th Annual Meeting of the Association for Computational Linguistics (ACL 1999), 152-158, Maryland, 1999.

[68] Lin, D., Using syntactic dependency as local context to resolve word sense ambiguity. In 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics (ACL/EACL 1997), pages 64-71, Madrid, 1997.

[69] Lin, D., Automatic retrieval and clustering of similar words. In COLINGACL'98, Montreal, 1998.

[70] Lin, D., Word sense disambiguation with a similarity-smoothed case library. Computers and the Humanities, 34(1-2):147-152, 2000.

[71] Jiang, Jay J., and Conrath, David W., Semantic similarity based on corpus statistics and lexical taxonomy. In: *Proceedings of ROCLING X*, Taiwan, 1997.

[72] Agirre, E. and Martinez, D. Knowledge sources for word sense disambiguation. In Matousek, V., Mautner, P., Moucek, R., and Tauser, K., editors, Proceedings of the Fourth International Conference TSD 2001, Plzen, Notes in Computer Science, 1-10, Berlin, 2001.

[73] Haynes, S., Semantic tagging usingWordNet examples. In Proceedings of Senseval-2, Second International Workshop on Evaluating Word Sense Disambiguation Systems, 79-82, Toulouse, 2001.

[74] Banerjee, Satanjeev, and Pedersen, Extended Gloss Overlaps as a Measure of Semantic Relatedness. In: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence IJCAI-2003, Acapulco, Mexico, 2003.

[75] Litkowski, K., Senseval: The CL research experience. Computers and the Humanities, 34(1-2):153-158, 2000

[76] Litkowski, K., Use of machine readable dictionaries for word-sense disambiguation in Senseval-2. In Proceedings of Senseval-2, Second International Workshop on Evaluating Word Sense Disambiguation Systems, 107-110, Toulouse, 2001.

[77] Mihalcea, R. and Moldovan, D., Word sense disambiguation based on semantic density. In Proceedings of the Coling-ACL'98 Workshop "Usage of WordNet in Natural Language Processing Systems", 16-22, Montreal, 1998.

[78] Gaustad, T., Linguistic Knowledge and Word Sense Disambiguation, PhD Thesis, 2004.

[79] Hirst, G., Semantic Interpretation and the Resolution of Ambiguity. Cambridge University Press, Cambridge, 1987.

[80] Dahlgren, K., MeDowell, J., and Stabler, Edward P., Knowledge representation for commonsense reasoning with text. Computational Linguistics, 15(3):149-170, 1989.

[81] Cotton, S., Edmonds, P., Kilgarriff, A., and Palmer, M., SENSEVAL-2: Second International Workshop on Evaluating Word Sense Disambiguation Systems, Toulouse, 2001.

[82] Mihalcea, R., Word sense disambiguation with pattern learning amd automatic feature selection, Journal of Natural Language and Engineering (JNLE), 8(4):343-358, 2002.

[83] Stetina, J., Kurohashi, S., and Nagao, M., General word sense disambiguation method based on a full sentential context. In Proceedings of the Coling-ACL'98 Workshop "Usage of WordNet in Natural Language Processing Systems", 1-8, Montreal, 1998.

[84] Collins, M., Head-Driven Statistical Models for Natural Language Parsing, PhD thesis, Computer and Information Science Department, University of Pennsylvania, Philadelphia, 1999.

[85] Martinez, D., Agirre, E., and Marquez, L., Syntactic features for high precision word sense disambiguation, In Proceedings of the 19th International Conference on Computational Linguistics (Coling 2002), 626-632, Taipei, 2002.

[86] Lin, D., Principle-based parsing without overgeneration, In 31th Annual Meeting of the Association for Computational Linguistics (ACL 1993), 112-120, Columbus, 1993.

[87] Dunning, Ted, Accurate methods for the statistics of surprise and coincidence. Computational Linguistics, 19(1):61-74, 1993.

[88] Weeds, J., A Comparison of Lin's Similarity Measure and Lee's Alpha-Skew Divergence Measure, 2002.

[89] Resnik, P., Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, Montreal, 1995.

[90] Hirst, G., and St-Onge, D., Lexical chains as representations of context for the detection and correction of malapropisms. In: Fellbaum, Christiane, ed., WordNet: An Electronic Lexical Database, MIT Press, 1998.

[91] Wu, Z. and Palmer, M., Verb semantics and lexical selection. In Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico, 1994.

[92] Patwardhan, S., Banerjee, S., and Pedersen, T., Using measures of semantic relatedness for word sense disambiguation. In Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, 2003.

[93] Alam, H. et al., Extending A Broad-Coverage Parser for a General NLP Toolkit, 2002. 454-460.

[94] Yarowsky, D., One sense per collocation. Proceeding of ARPA Human Language Technology Workshop, Princeton, New Jersey, 266-271, 1993.

[95] Yarowsky, D., Decision lists for lexical ambiguity resolution: application to accent restoration in Spanish and French. Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico, 88-95, 1994.

[96] Yarowsky, D., A comparison of corpus-based techniques for restoring accents in Spanish and French text. Proceedings of the 2nd Annual Workshop on Very Large Text Corpora. Las Cruces, 19-32, 1994.

VITA

Basak Mutlum completed the high school in Eskisehir Anatolian High School, Eskisehir, in 1998. She received her B. Sc. degree in Computer Engineering from Middle East Technical University (METU), Ankara, in 2003. As her senior design project she has been involved in a platform independent system monitoring project at METU. Since 2003, she is in the M. Sc. program in the Computer Engineering Department of Koc University as a teaching and research assistant.