ALL-WORDS WORD SENSE DISAMBIGUATION IN TURKISH

SİNAN AKÇAKAYA
B.S., Computer Engineering, Izmir Institute of Technology, 2007

Submitted to the Graduate School of Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science
in
Computer Engineering

IŞIK UNIVERSITY
2019

IŞIK UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

ALL-WORDS WORD SENSE DISAMBIGUATION IN TURKISH

SİNAN AKÇAKAYA

APPROVED BY:

Prof. Dr. Olcay Taner YILDIZ (Thesis Supervisor)	Işık University
Assist. Prof. İlknur KARADENİZ EROL	Işık University
Assoc. Prof. Arzucan ÖZGÜR	Boğaziçi University

APPROVAL DATE: 06/09/2019

ALL-WORDS WORD SENSE DISAMBIGUATION IN TURKISH

Abstract

Word sense disambiguation (WSD) is the identification of the meaning of words in context in a computational manner. The main subject of this study is to implement and compare the WSD results of various supervised classifiers (Naive Bayes, K Nearest Neighbor, Rocchio and C4.5) in all-words setting. To this end, we have constructed an all-words sense annotated Turkish corpus, using traditional method of manual tagging. During the annotation, a pre-built parallel treebank (aligned from Penn Treebank) has been tagged with the senses of Turkish Language Institutions dictionary. The approach of annotating a treebank allowed us to generate a full-coverage resource, in which syntactic and semantic information merged.

In the WSD evaluations, three distinct experiments have been organized to determine the effect of using different feature sets on the disambiguation performance. First experiment has been conducted with a simple feature set that includes the fundamental local features. In the second experiment, the initial feature set has been augmented with several effective morphological features, and in the third one, the feature set has further been extended with the syntactic features. Our test results show that all classifiers have achieved better results in parallel to growing feature set. Additionally, integration of syntactic features has proved to be useful for WSD.

Keywords: All-words WSD, Natural Language Processing, Syntactic Features, Supervised Learning

TÜRKÇE TÜM SÖZCÜKLER İÇİN ANLAM BELİRSİZLİĞİNİ GİDERME

Özet

Anlam belirsizliğini giderme, kelimelerin bağlam içerisindeki anlamının hesaplamalı yöntemlerle belirlenmesidir. Bu çalşmanın ana konusu, çeşitli gözetimli sınıflandırma metodlarını (Naive Bayes, K Nearest Neighbor, Rocchio ve C4.5) Türkçe bir metindeki tüm sözcüklerin anlam belirsizliğini gidermek için uygulamak ve elde edilen sonuçları karşılaştırmaktır. Bu amaçla, geleneksel elle işaretleme yöntemini kullanarak Türke tüm sözcükler için bir derlem oluşturduk. Etiketleme esnasında, önceden çözümlenmiş (Penn Treebank) ve Türkçe'ye uyarlanmış paralel bir derlem Türk Dil Kurumu'nun sözlüğündeki anlamlarla etiketlenmiştir. Çözümlenmiş bir derlemin etiketlenmesi bize içerisinde anlamsal ve sözdizimsel bilginin harmanlandığı tam kapsamlı bir derlem meydana getirme imkanı tanımıştır.

Anlam belirsizliğini giderme testlerinde farklı özellik kümelerinin performansa olan etkisini saptamak için üç ayrı deney hazırlanmıştır. Birinci deney, temel lokal özellikleri içeren yalın bir özellik seti ile yapılmıştır. İkinci deneyde bu yalın küme çeitli morfolojik (biçimbilimsel) özelliklerle genişletilmiştir. Üçüncü deneyde ise sözdizimsel özelliklerin eklenmesiyle daha da kapsamlı bir özellik kümesi oluşturulmuştur. Deney sonuçları tüm sınıflandırma yöntemlerinin özellik kümesinin genişletilmesine paralel olarak daha yüksek performans değerleri elde ettiğini göstermektedir. Ayrıca, sözdizimsel özelliklerin entegrasyonunun anlam belirsizliğini gidermede faydalı olduğu gösterilmiştir.

Anahtar kelimeler: Anlam Belirsizliğini Giderme, Doğal Dil İşleme, Sözdizimsel Özellikler, Denetimli Öğrenme

Acknowledgements

I would like to thank all my masters in my education journey. This study is the fruitage of their joint effort.

My valuable master Prof. Olcay Taner Yıldız deserves special thanks. From the beginning to the end of this study, he has not only been a great advisor but also good friend. I appreciate his competence and patience.

And finally, I want to thank my parents for their endless support. None of this would have been possible without support of them.

To my Family…

Table of Contents

List of Tables

List of Figures

List of Abbreviations

WSD	**W**ord **S**ense **D**isambiguation
ML	**M**achine **L**earning
NLP	**N**atural **L**anguage **P**rocessing
MT	**M**achine **T**ranslation

Chapter 1

Introduction

Word sense disambiguation (WSD) is a historical natural language processing (NLP) task that aims to identify the meaning of words in context with an automatic method. WSD can be seen as a classification task in which word senses are the classes. Word occurrences are assigned to a class (sense) with the help of information elicited from the context and other knowledge resources. Although the task can associate a word with more than one sense, oftentimes it is expected to select only the most appropriate sense. For instance, consider the word *bear* in the following sentence:

I couldn't bear watching my friend get bullied and had to step up to help him.

It is a polysemous verb that can assume several meanings depending upon the text in which it occurs (see Table 1.1).

Sense Number	**Sense Definition**
1	Carry
2	Take responsibility
3	Give birth
4	Manage to tolerate a situation
5	Turn and proceed in a specified direction
6	Strongly dislike

Table 1.1: Possible senses of the verb *bear*.

The WSD process finds the most likely sense of a word among the possible set of senses (i.e., classes) by applying an automatic classification method. For example, the word bear in the above sentence should ideally be tagged with the 4th class: *Manage to tolerate a situation.*

The usefulness of text disambiguation in an applicative perspective is quite obvious. There are a number of end-to-end language technology applications (e.g., information retrieval, machine translation, content analysis, etc.) which might benefit from WSD. For instance, the English word ”pen” can be translated in Turkish as ”kalem”, ”kümes”, ”dişi kuğu”, etc., depending on the context. In the problem of machine translation, there are thousands of similar cases where automatic translators need more than the surface appearance of words.

Even though the field has extensively been examined from many different angles (especially for English), WSD has not yet demonstrated real benefits in vivo. This is a consequence of the current insufficient performance of WSD and there are still ongoing endeavors (e.g., devising more advanced learning models, exploring effective feature sets, domain-driven disambiguation, etc.) towards development of higher performance systems. Concerning the Turkish language, many of the topics within the area are still open to research, as it differs from English by its agglutinative nature. As a matter of fact, knowledge acquisition is still an important issue, because of the scarcity of NLP resources, such as labeled corpora and semantic networks. We believe that construction of this kind of resources and using them in Turkish WSD experiments can provide significant insights in our way to language understanding systems.

In this study, we present an all-words sense annotated Turkish corpus. This dataset has been built on a ready-made parallel treebank [1] which is aligned from the well-known Penn Treebank II corpus [2]. Our corpus has the advantage of ready-to-use root words and other morphological information, as it has been analyzed morphologically in a distinct research [3]. Main goal of this work is to implement and compare the disambiguation results of various supervised classifiers (Naive Bayes, KNN, Rocchio, C4.5) on this dataset. In the evaluations, different feature sets have been supplied to the classifiers to determine the most effective features that give the topmost sense differentiation results. In particular, we have aimed to explore the impact of syntactic features on the performance of Turkish WSD systems.

The remaining of the dissertation is organized as follows: In Section 2, we discuss WSD variants and main approaches applied to the problem. Next, we give a review of the literature in Section 3. Later in Section 4, we share our experience on hand tagging of the senses in the Turkish Treebank and explain characteristics of the employed tool. Section 5 describes the feature sets applied in the experiments. Section 6 introduces evaluation methodology and provides experiment results of different algorithms. Finally, we comment on the experimental outcome and conclude with the lessons learned to achieve better results in Section 7.

Chapter 2

Approaches to WSD Problem

In this chapter, we introduce two variants of the WSD and tackle main approaches applied to the problem.

2.1 WSD Variants

WSD can be categorized into two variants: lexical sample and all-words. In lexical sample task, system disambiguates restricted set of target words that has been determined previously. Supervised machine learning (ML) algorithms are often employed in this setting, as it is likely to create a sufficient size of training set for the small number of target words. On the other hand, all-words WSD tasks are supposed to remove ambiguities from all open-class words (nouns, verbs, adjectives and adverbs) in a document. This task calls for extensive lexical knowledge resources. Consequently, all-words disambiguation is attacked with either hybrid methods or supervised tactics backed off with the most frequent sense (MFS) heuristic.

2.2 WSD Approaches

Approaches to WSD are commonly distinguished according to the origin of knowledge used in providing a sense choice. Methods that rely on corpora of texts, either annotated with senses or raw (unlabeled), are termed as corpus-based. Systems which exploit the knowledge in dictionaries, wordnets, lexicosemantic relations (hypernym, hyponym, nominalization, etc.), collocations, and so on, are termed knowledge-based (or dictionary-based). Figure 2.1 displays the types of approaches for WSD. In what follows, we summarize these approaches.

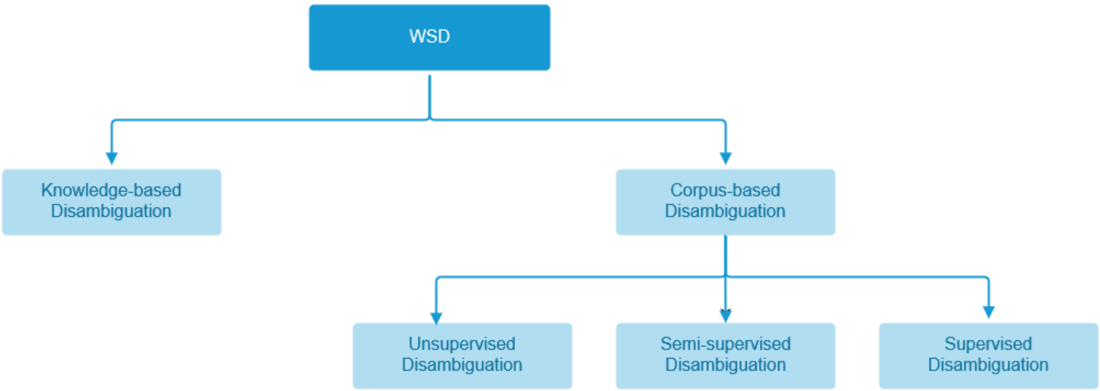

WSD
Knowledge-based Disambiguation
Corpus-based Disambiguation
Unsupervised Disambiguation
Semi-supervised Disambiguation
Supervised Disambiguation

Figure 2.1: Types of approaches for WSD.

2.2.1 Knowledge-Based Disambiguation

Knowledge-based techniques make use of the structure or content of the lexical knowledge bases, such as dictionary definitions, thesauri, ontologies and domain labels, without using any corpus evidence. The performance of these methods is generally beaten by corpus-based techniques. However, they are superior from another perspective. They are applicable to all words in a running text, in contrast to the corpus-based methods, as they do not require sense tagged corpus. Methods using contextual overlap between the target word and its dictionary definitions (Lesk algorithm) is one of the first approaches in this kind. Based on the same principle, the original algorithm has been extended in various works (see, e.g., [4] and [5]). Some other knowledge-based systems rely on selectional preferences (refer e.g., [6]). These systems capture several relations between words, and performs sense assignment by ruling out the senses that violate the learned semantic constraints. Additionally, beginning with the WordNet [7] style lexicons, a number of graph based approaches have been developed which exploit the structural properties in semantic graphs to be able to infer the correct senses of words in a given text (see, e.g., [8] and [9]).

2.2.2 Corpus-Based Disambiguation

Corpus-based techniques are further categorized in terms of the degree of supervision, that is, the proportion of sense-labeled data to plain data:

(1) Supervised systems exclusively use sense-annotated training data to build a word expert (i.e., classifier).

(2) Semi-supervised or (minimally supervised) systems employ both labeled and unlabeled (raw) data together in different proportions.

(3) Unsupervised methods completely eschew annotated data and work from unannotated plain corpora.

2.2.2.1 Supervised Disambiguation

Supervised WSD requires a training phase before the resolution of word ambiguity. In the training phase, ML methods are used to learn a classifier from the training set, i.e., set of examples in which target words are encoded as vectors of features (such as part-of-speech (POS) tags, argument-head relations, root forms, syntactic relations) as well as their sense classification tags. During the test phase, classifier associates words with their appropriate senses, so that we can get the working accuracy of our model. Typical workflow of a supervised WSD system is shown in Figure 2.2.

Generally, supervised WSD have obtained better results than the other methods. However, they require training sets of adequate size to achieve high performance. Thus, the lack of widely available large-scale tagged corpora (so-called knowledge acquisition bottleneck) is the main weakness of supervised methods. We briefly present here the ML algorithms used in our evaluations.

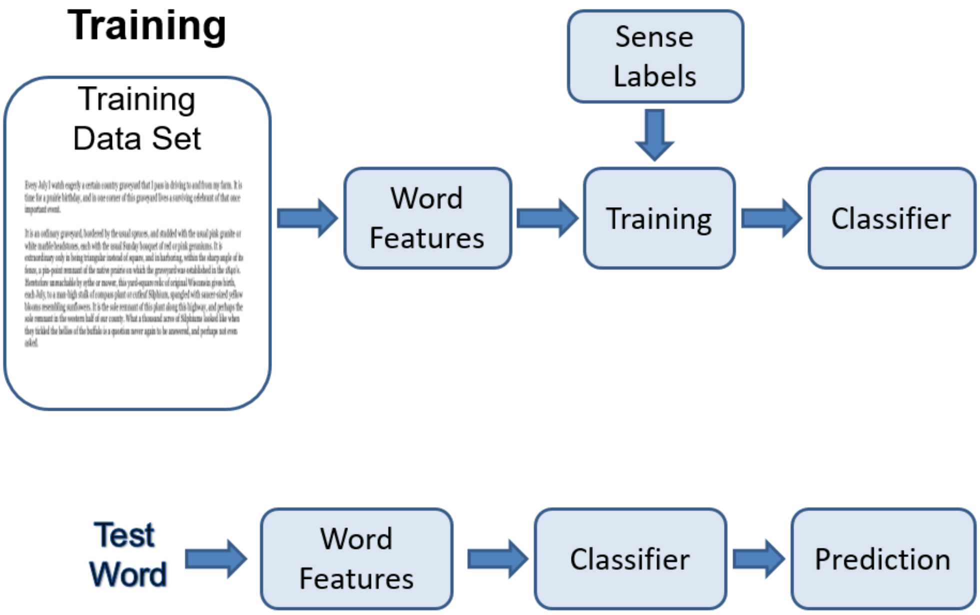


Figure 2.2: Supervised WSD workflow.

2.2.2.1.1 k-Nearest Neighbor

In the k-Nearest Neighbor (kNN) algorithm, the model is built by retaining all instances in memory. In the classification of a new instance, algorithm finds out the k nearest (most similar) samples among the pre-stored set of instances. Then, the instance is predicted to belong to the most frequent sense among the k-nearest neighbors.

Let X be a test instance that we want to categorize. In order to obtain the k-nearest neighbors of X, distance between the X and every pre-stored training instance is calculated. There are several ways to calculate the closeness (or distance) between two samples. In our experiments, we used Euclidian distance to measure similarity, defined as:

$$\Delta(X,Y) = \sum_{j=1}^{m} w_j \delta(X_j, Y_j) \tag{2.1}$$

where m is the number of features, w_j is the weight of the j-th feature and $\delta(X_j, Y_j)$ is the distance, that is 0 if $X_j = Y_j$ and 1 otherwise.

Optimal value of the k parameter can be established experimentally. Exemplar weights (available for only k values greater than 1) can also be estimated which could lead to better results.

2.2.2.1.2 Rocchio Algorithm

The basic idea in this algorithm is to construct one prototype instance per class (i.e., the sense), using the collection of training items. When testing new samples, Euclidian distance between the test sample and the class prototypes are calculated, and class of the closest prototype is chosen as the sense of the sample.

The prototype vector of a class is found by calculating the mean of each attribute according to their individual distributions. For discrete attributes, the value of the item with the highest frequency will be the mean for the distribution. If the attribute is continuous, the mean is the usual average.

2.2.2.1.3 Naive Bayes

Naive Bayes is a probabilistic classification technique based on Bayes theorem with the assumption of independence among features. It consists in the calculation of the conditional probability of each possible sense of a word given the observed context features. The sense category for which the estimated probability is highest, is selected as the most appropriate sense.

Suppose that we want to identify the meaning of a word (represented with m features $f_1, f_2, \cdots, f_m$) that has N possible senses. For the ith $(i < N)$ possible sense S_i, the conditional probability is calculated by the following formula:

$$P(S_i|f_1, \cdots, f_m) = \frac{P(f_1, \cdots, f_m|S_i)P(S_i)}{P(f_1, \cdots, f_m)} \tag{2.2}$$

In this formula we can discard the denominator, because it does not affect the comparison result. Since the Naive Bayes approach assumes the independence of the features, we can reformulate the equation as follows:

$$P(S_i|f_1, \cdots, f_m) = P(S_i)\prod_{j=1}^{m} P(f_j|S_i) \tag{2.3}$$

The probabilities $P(S_i)$ and $P(f_j|S_i)$ are the probabilistic parameters of this model. First one is estimated as the number of examples of S_i over the total number of examples in the training set. The maximum likelihood estimation for the $P(f_j|S_i)$ is the ratio between the number of sense S_i examples whose j-th feature is equal to f_j and the number of examples of S_i.

2.2.2.1.4 C4.5 Algorithm

C4.5 is a popular algorithm for generating decision trees. A decision tree is used to describe classification rules with a multi branching tree structure. During the training phase, training data is partitioned in a recursive manner to build the model (i.e., decision tree). In the classification of new samples, decision tree is traversed, and a prediction is made when a leaf node is attained. Each internal node in the tree represents a test on an attribute value, and each link represents a result of the test.

2.2.2.2 Semi-supervised Disambiguation

Manual annotation is the most common and reliable way for creating sense annotated training data. However, manual creation of sense-annotated corpora is a high-priced and laborious effort. A number of methods (often called semi-supervised), which use unlabeled corpora as well as a partial amount of sense-tagged data, have been proposed to overcome the manual-annotation bottleneck.

Bootstrapping [10] usually starts from a few annotated data and runs a set of one or more classifiers on a vast amount of unlabeled data. As a result of iterative applications of the classifiers, the annotated corpus augments increasingly.

Another idea in this respect, views web as a raw corpus and purposes to annotate web data with the aid of monosemous relatives. First, one or more search phrases are determined from a dictionary which uniquely identifies the sense of a word. Then, Web is searched for the expressions found in the first step and text snippets are retrieved. Finally, occurrences of phrases are replaced with the word in the snippets. Result of this process is an annotated corpus of the word sense that we have focused on it. A similar method was proposed by [11] for the construction of topic signatures.

2.2.2.3 Unsupervised Disambiguation

Unsupervised techniques do not carry out sense assignment for the words, but rather perform sense discrimination, that is, grouping word occurrences into a number of clusters, where clusters consist of items with the same meaning. Main idea behind the unsupervised disambiguation is that the occurrences of words with the same sense tend to appear with similar surrounding words. These approaches have potential to relieve the knowledge acquisition bottleneck, as they do not rely on external knowledge resources like dictionaries, wordnets, sense-tagged training text, etc.

Chapter 3

Related Work

3.1 The SENSEVAL/SEMEVAL Competitions

Senseval competitions are ongoing series of workshops which aim at evaluating semantic analysis systems. The first three evaluations, namely: Senseval-1 (1998), Senseval-2 (2001) and Senseval-3 (2004) have mostly focused on WSD. Beginning with the fourth edition (Semeval-2007), the organization has been renamed as Semeval, given the presence of tasks not necessarily related to WSD. However, the workshops still deserve attention for WSD researches, since they lead to the periodic release of datasets of high value.

The Senseval workshops are the best reference to WSD works. Several kinds of tasks, including explicit (lexical-sample, all-words) and implicit WSD for various languages have been proposed by the systems submitted to these competitions. Unfortunately, there exists no recorded study on all-words WSD for the Turkish language. We will outline here some of the all-words systems for English.

Senseval-2 test data consisted of 2,473 words of running text from the Penn Treebank II. The WordNet 1.7 lexicon was adopted as a sense inventory. MFS baseline accuracy was equal to 57% for this task. Winner of the Senseval-2 English all-words task is the SMUaw system [12] with a 69% performance. It is a supervised system with two main components: memory-based learning (MBL) and pattern learning. If there are enough samples for a target word, MBL is applied with an active feature selection, that is, only the features that improves the performance are kept for each word. Otherwise, pattern learning is applied. Patterns (each token is represented by its root form, part-of-speech tag, actual sense and hypernym) are obtained from SemCor [13], WordNet definitions and GenCor [14]. These modules were preceded with a pre-processing stage where named entities were detected and collocations extracted. The system assigned the most frequent sense when both components were not feasible.

IRST-DDD [15] system is the best unsupervised one with an achievement of 10th position after nine supervised systems in Senseval-3. Its accuracy was 58.3%, whereas the MFS baseline attained 60.9%. This system adopted domain-driven disambiguation approach in which sense of a target word is chosen based on a comparison of the estimated domain of the context of the target word to domains of its senses. Domain information was represented in terms of domain vectors and they were estimated with an unsupervised method (Gaussian Mixture) that describes the frequency distribution of words inside a large-scale corpus. In particular, they used BNC corpus [16] and exploited a version of WordNet augmented with domain labels.

Both a fine-grained and a coarse-grained disambiguation exercise were organized at Semeval-2007 to understand the impact of sense granularity on WSD performance. The best participant in the coarse-grained task, namely, the NUS-PT system [17], attained an 82.5% precision. This performance is way ahead of the former ones, showing that major obstacle to high performance WSD is the fine granularity of sense inventories. In this task, test data (2269 sense tagged samples) was labeled according to coarse-grained version of WordNet 2.1 released by task organizers. The system adopts a supervised approach based on support vector machines (SVM) algorithm, using the knowledge sources of local collocations, POS and surrounding words. Training instances were gathered from the SemCor, English-Chinese parallel corpora and the DSO corpus [18]. If there was no training data for a word, it was tagged with the first entry in the WordNet.

3.2 Turkish WSD Works

In parallel to increasing concern over Turkish NLP, there have been some efforts in WSD. The single participant system [19] for the Turkish Lexical Sample Task, held in Semeval-2007, is one of the efforts in this direction. This system uses Naive Bayes method to build word experts, using the syntactic attributes. 5,385 samples of 26 highly ambiguous words from the METU-Sabanci Turkish Treebank [20] has been sense tagged with Turkish Language Institutions (a governmental organization-abbreviated as TDK) dictionary in data preparation phase. Approximately 70% of the samples were reserved for training classifiers and the remaining as evaluation data. The precision results that system achieved are 65%, 56% and 57% for the nouns, verbs and others respectively.

The study reported in [21], aims to investigate the effect of diverse set of features on WSD performance. They utilized the Turkish Lexical Sample Dataset (TLSD) [22], which includes noun and verb sets, each of which has 15 words with high polysemy degree. This dataset has at least 100 samples for each chosen word. Various supervised methods (Naive Bayes, MBL, decision tree, functional tree, SVM), have been applied on the TLSD using two settings: (1) separate employment of collocational and bag-of-words (BoW) features, (2) joint setting of two feature sets. Functional trees gave the foremost score in both nouns and verbs for collocational features. Average accuracy results were 73.47% and 67.26% for nouns and verbs respectively. In the employment of bag-of-words, decision tree approach (J48) yielded best scores with the 59.42% and 46.69% outcome of nouns and verbs. According to reported results, collocational features are more effective than BoW features in the resolution of word ambiguity. However, joint involvement of the attribute sets with a feature selection exceeded the achievement of collocational attributes. In this case, SVM was the top-ranking algorithm with an outstanding performance of all word types (78.91% for nouns and 74.03% for verbs). The authors claimed that the appropriate selection of features can contribute more than the contribution of using different algorithms, given the experiment results.

Yet another study [23] on this subject, suggests that using predictive features and eliminating redundant ones increase the disambiguation performance. They have run a feature selection algorithm using correlation based method on the collection of TLSD. The results present the effective local features for ambiguous noun and verb sets when window size is taken four. As stated by the authors, the efficient features are mostly placed between $W_{-}2$ and $W_{+}2$. Also, roots and the major POS in both directions play an important role in the disambiguation of word senses.

Chapter 4

Dataset Construction

4.1 Input Corpus

Nowadays, most of the well-known corpora had not only the words and their senses. They have been subject to certain POS tagging and syntactic parsing efforts. Uncovering the syntactic structure of a sentence has clearly many applications in NLP. For instance, in WSD, each word is represented as a list of features that describe the occurrence under the surrounding context (sentence, paragraph or full document) to make it a suitable input for an automatic method. A great majority of these features come from the applications such as, POS tagging, morphological analysis and syntactic parsing.

Besides, lemmatization (a subtask in morphological analysis) is a must before the semantic annotation process. Turkish is an agglutinative language in which words are formed by attaching derivational and inflectional suffixes to the root words. During the word formation some letters may disappear or may transform to the other characters. Therefore, surface form of the words must be reduced to their base form (called lemma) to be able to extract candidate senses from a reference inventory.

As the preprocessing of the input text (POS tagging, lemmatization, morphological decomposition, chunking etc.) is commonly required in WSD, we have annotated a ready-made English-Turkish parallel treebank [1], aligned from a subset of the Penn Treebank II corpus. This parallel treebank covers 9,560 sentences with a maximum length of 15 tokens, including punctuation. Following the alignment, our treebank has further been analyzed morphologically in a distinct study of tree-based statistical machine translation (SMT) [3].Thus, it has the advantage of ready-to-use root words and other morphological and grammatical information. In the following subsections, we present prior treebank alignment and morphological analysis efforts.

4.1.1 Alignment

In the work of [1], authors report that they have converted English parse trees into equivalent Turkish trees by applying several transformation heuristics. Firstly, they permuted subtrees (only children of an ancestor node could be rotated among themselves) in accordance with the Turkish sentence structure. Then, leaf tokens were replaced with the most synonymous Turkish counterparts. Figure 4.1 illustrates the syntactic parse of an original Penn Treebank sentence "Ms. Haag plays Elianti."

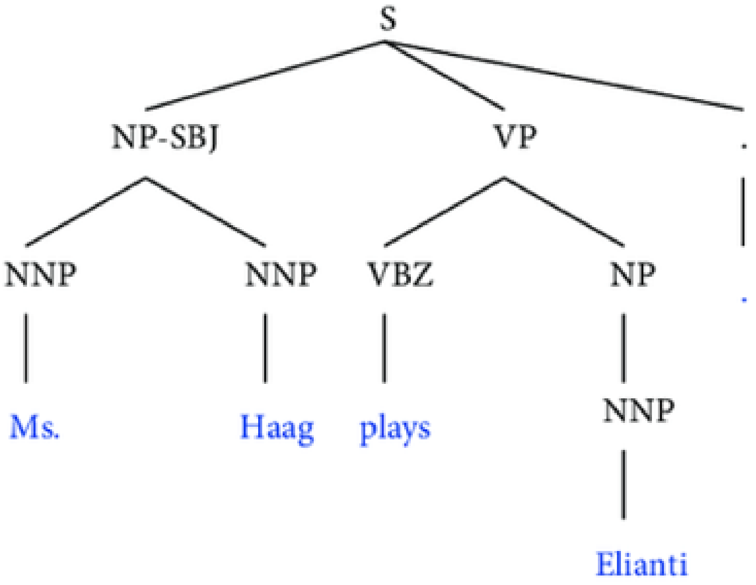


Figure 4.1: Syntactic parse of a Penn Treebank sentence: "Ms. Haag plays Elianti.".

Through the process, no nodes were added or deleted, and they used the original Penn Treebank tagset, with the exception of a new tag: **NONE** (used when direct translation from English to Turkish was not possible). In the aligned form of the above sentence, VBZ and NP nodes are permuted to reflect the syntactic transition from the English side to Turkish side. Figure 4.2 shows the Turkish translated form of the sentence shown in Figure 4.1.

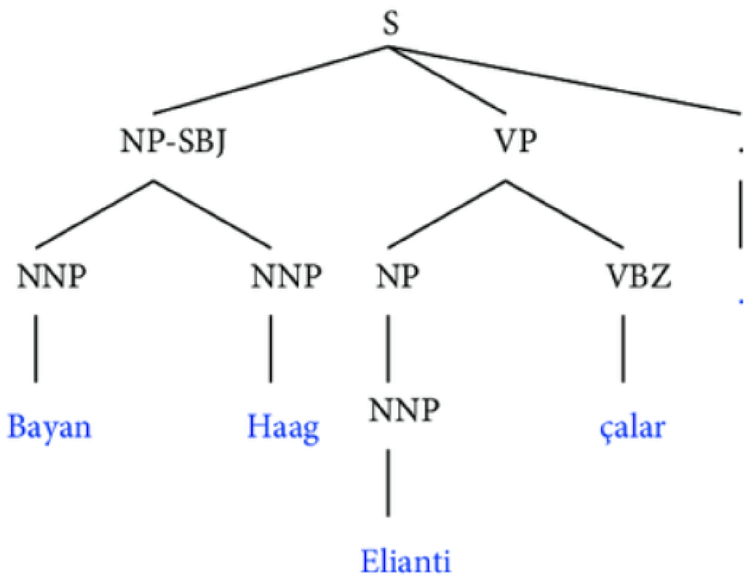


Figure 4.2: Turkish translated form of the Penn Treebank sentence: "Ms. Haag plays Elianti.".

For detailed information about the Penn Treebank corpus and syntactic tagset, the reader can refer to [2].

4.1.2 Morphological Analysis

As mentioned before, in Turkish, words can be in several variants because of the agglutinative word structure. In order to obtain possible senses of a word, it must be reduced to its base form from the current morphological variant. Our input corpus has been enhanced with the root words and other morphological information within the scope of an English to Turkish MT study, recorded in [3].

Despite sufficient availability of morphological processing tools for Turkish, a morphological parser was built within the field of the activity. Each output of the parser begins with the root word and then POS tag is given. These are succeeded by a set of morphological features (such as case markers, person and possessive agreements, polarity, tenses), each separated with a + sign. For example, consider the Turkish statement illustrated in Figure 4.3. It is taken from our parallel treebank, and corresponds to the following Penn Treebank sentence:

Some companies are starting to tackle that problem.

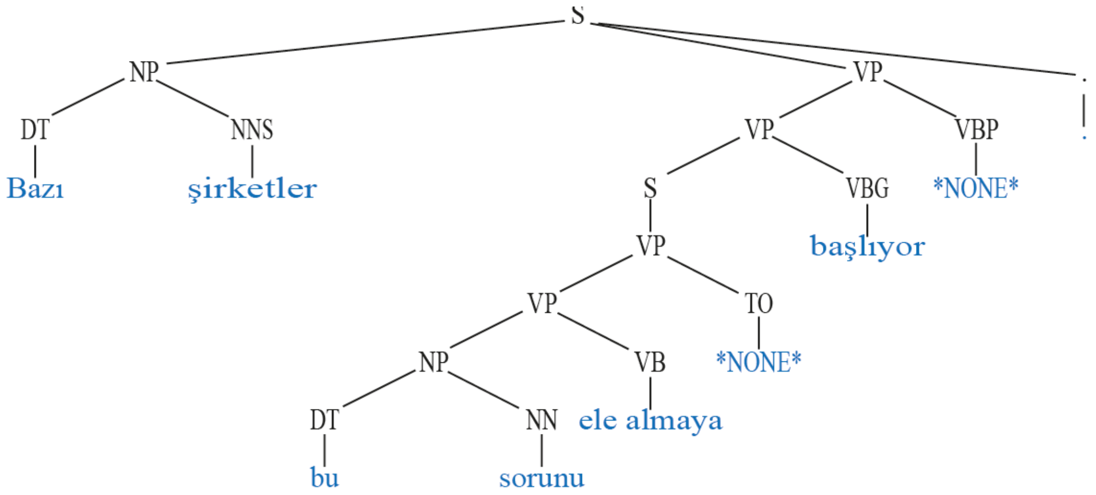


Figure 4.3: Turkish translated form of the Penn Treebank sentence: ”Some companies are starting to tackle that problem.”.

Table 4.1 lists three possible analyses produced by the analyzer for the word *sorunu* in the above Turkish statement.

Morphological Analyses	**English Gloss**
sorun + NOUN + A3SG + PNON + ACC	the problem/that problem
sorun + NOUN + A3SG + P3SG + NOM	his/her problem
soru + NOUN + A3SG + P2SG + ACC	your question

Table 4.1: Possible morphological analyses of the word *sorunu*.

This parser adopts the tagset and the morphological representation used in a past treebank compilation project [20]. The interested reader can refer to [20] for an in-depth information about the Turkish morphological features.

Since the output of this morphological analyzer is ambiguous, they had to resolve this parsing ambiguity. In that work, morphological disambiguation was manual, in contrast to the automatic analysis. Human annotators selected the correct morphological parse from the multiple potential analyses returned from the automatic parser. For example, correct parse for the word *sorunu* in the above sentence is represented with: sorun + NOUN + A3SG + PNON + ACC.

4.2 Annotation Methodology

In the sense annotation phase, overall process has been carried out with respect to the TDK dictionary, using traditional method of manual tagging. Annotation process has been performed by five taggers majoring in the computer engineering. At the beginning of the activity, workload was distributed among them. Each annotator has engaged in labeling distinct set of sentences. However, when it comes to distinguish too fine-grained senses, they have asked other taggers to select the most appropriate sense for those items. The subtle items, for which disagreements have arisen through the process, have been assigned with the sense given by majority of the five human annotators. After the entire corpus has been tagged, annotators have been asked to label common 250 sentences from the corpus to evaluate the quality of annotation task performed so far. We have measured 83% inter-tagger agreement on 2,054 words, which are organized in the common test set.

4.3 Sense Inventory

Sense inventories (dictionaries, wordnets) are essential for semantic annotation, as they partition the range of meaning of words into its senses. In contrast to a common dictionary, which provides definitions for words and generally lists them in alphabetical order, in wordnets, all open-class words are grouped into sets of cognitive synonyms called synset. Furthermore, synsets are interlinked by means of conceptual (antonym, hypernym, hyponym etc.) and lexical relations. So, wordnets are often considered one step beyond traditional dictionaries and thesauri. After the 2000s, the NLP community widely adopted WordNet [7] instead of classic dictionaries. WordNet is a large lexical database of English, which is created and maintained at Princeton University.

Despite the usefulness of wordnets over the classic dictionaries, since we dont have large-enough Turkish wordnet, the dictionary of the TDK has been utilized during the process. For instance, Turkish part of BalkaNet (wordnets for six European languages) [24] contains only 11,628 synsets with 16,095 literals (members of synsets) in them; whereas the dictionary of TDK is a collection of 92,371 distinct lemmas (dictionary form of a word) organized in 121,602 sense entries. If we neglect 75,072 monosemous words in the dictionary, average number of sense entries per lemma is calculated as 2.69. Figure 4.4 [22] shows distribution of the polysemy degree in the dictionary of TDK.

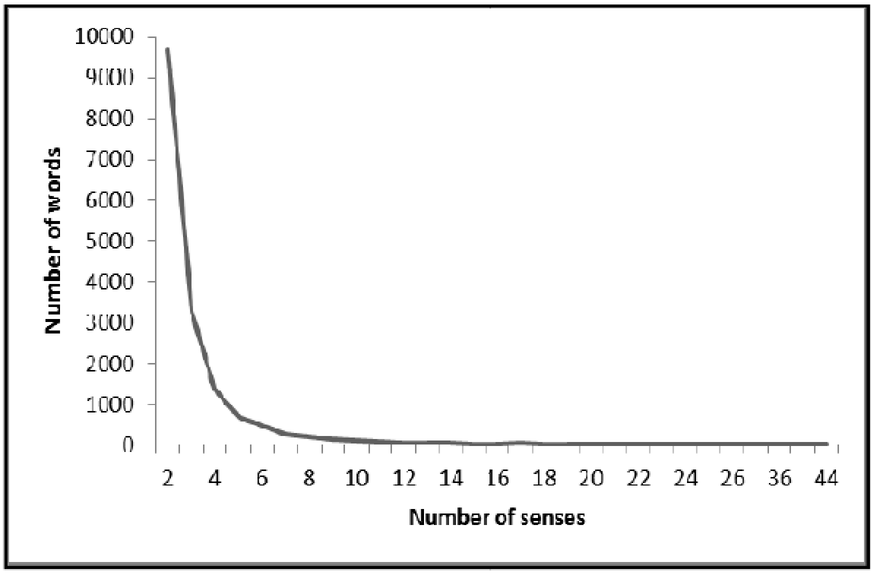
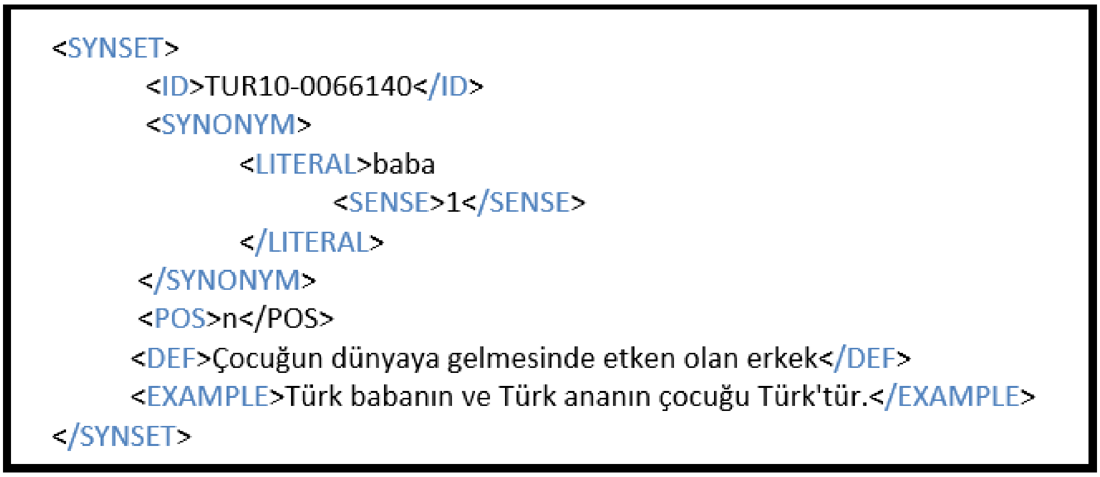


Figure 4.4: Distribution of polysemy degree in the TDK dictionary.

We store the TDK dictionary in XML format that is fairly similar to the BalkaNets. In our format, units that constitute the vocabulary are the possible senses of the words. We named these units as synset as in the domain of the wordnets but, our synsets are not merged or interlinked with other synsets. Indeed, we have not made extra processing on the original dictionary; just transformed into a format which resembles the BalkaNet style. The structure of a sample synset is reported in Figure 4.5.



```
<SYNSET>
        <ID>TUR10-0066140</ID>
        <SYNONYM>
                <LITERAL>baba
                        <SENSE>1</SENSE>
                </LITERAL>
       </SYNONYM>
       <POS>n</POS>
      <DEF>Çocuğun dünyaya gelmesinde etken olan erkek</DEF>
      <EXAMPLE>Türk babanın ve Türk ananın çocuğu Türk'tür.</EXAMPLE>
</SYNSET>
```

Figure 4.5: Structure of a sample synset for the word *baba*.

Each entry in the dictionary is enclosed by <SYNSET> tags. Synset members are represented as literals and their sense numbers (order of the sense among the all senses of the word). In contrast to the BalkaNet, where synonym literals are joined in a synset, our synsets can have only one literal. <ID> shows unique identifier given to the synset. <POS> and <DEF> tags denote POS and definition. As for the <EXAMPLE> tag, gives a sample sentence for the synset.

4.4 Basic Data Format

Through the above-mentioned labours (translation-morphological analysis) and the disambiguation work explained here, common materials were treated by remaining highly faithful to standard Penn Treebank notation of syntactic bracketing in the backend. Each of these successive studies enriched the documents in terms of the respective application goals, based on the previous one. For instance, consider the following parallel statements presented in Figure 4.1 and Figure 4.2.

Ms. Haag plays Elianti.
Bayan Haag Elianti çalar.

In the Figure 4.1, the word *plays* is associated with the VBZ tag. VBZ (verb, 3rd person singular present) is a kind of POS tag. POS tags are assigned to a single word according to its grammatical role in the sentence. As following the standard Penn Treebank notation, *plays* is represented as follows: (VBZ plays).

This word corresponds to the *çalar* in the Turkish side (see Figure 4.2). Table 4.2 shows the basic data format for the word *çalar* throughout the successive activities. Note that extensions to the data are made on the leaf nodes by adding relevant info between curly braces.

Activity	**Data Format**
Penn Treebank	(VBZ plays)
Translation (Alignment)	(VBZ {turkish=çalar} {english=plays})
Morphological Analysis	(VBZ {turkish=çalar} {english=plays} {morphologicalAnalysis=çal+VERB+POS+AOR+A3SG})

Semantic Annotation	(VBZ {turkish=çalar} {english=plays}
	{morphologicalAnalysis=çal+VERB+POS+AOR+A3SG}
	{semantics=TUR10-0148580)

Table 4.2: Basic data format for the word çalar throughout the activities.

Translation step extends the treebank data with the Turkish equivalents of the words. Morphological analysis adds root and morpheme information for the construed units. Eventually, tagging the words with their semantic definitions enhances the file content with the unique identifier given to that sense in our reference dictionary.

4.5 Annotation Tool

To assist human annotators, we built a custom application (written in Java) designed for semantic annotation of all words in a POS tagged and morphologically analyzed sentence. This application is a user-friendly interface tool that enables visual browsing and tagging. Our application has been placed as a part of the Işık University NLP Toolkit, collection of in-house tools designed for NLP studies. Thus, we could use same infrastructure with the prior treebank alignment and morphological analysis tasks.

The current version of the application is designed for the import of text files that adhere to our Penn Treebank style data format. Once a pre-processed (that is, translated and morphologically analyzed) sentence has been imported into the semantic editor (there exists separate windows for each function provided by the tool), the human annotator is presented with the visualized syntactic parse tree of that sentence. On the semantic editor, annotators can only click on leaf nodes (words). They are not allowed to make any changes like rotating or deleting nodes. When a word is selected, a drop-down list is shown, in which all available TDK entries of the current item are listed. Figure 4.6 illustrates a screen shot from the application interface, when tagging the verb *çalar* in the sentence shown in Figure 4.2.

Each sense result shown to the users is populated with its POS and a sample sentence, if available in the TDK dictionary. This becomes a considerable aid to the users to decide which sense to assign to a target word. Right after the selection of the most proper sense definition, the drop-down list is hidden, and id

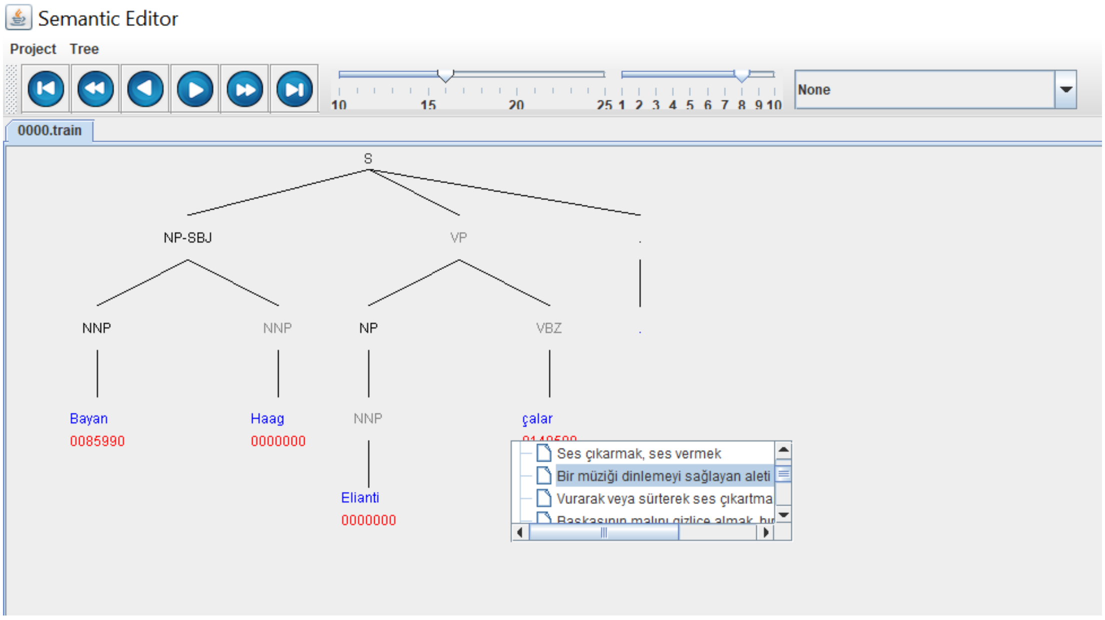


Figure 4.6: Screen shot from the application interface.

of the synset that corresponds to the submitted sense definition is displayed just under the word.

In all-words annotation tasks, list of words to be tagged, and therefore candidate senses, are unpredictable. Our application handles sense extraction on behalf of annotators by querying lemma of the selected word in the lexicon. For instance, reconsider the aligned Turkish sentence given in section 4.1.2.

Bazı şirketler bu sorunu ele almaya başlıyor.

The word *sorunu* in this sentence can morphologically be decomposed in a three different ways as shown in the Table 4.1. When this word is selected for sense tagging, our system pulls the senses of either *soru* (question) or *sorun* (problem) according to the hand-made morphological disambiguation performed in advance of the sense annotation. As it is understood, the accuracy of the preliminary steps had a powerful influence in the success of annotation we did, but we could make necessary corrections when required.

4.6 Features and Statistics of the Corpus

Our corpus is not an isolated words library. Commentators are asked to select the most appropriate sense for all words in a given sentence. As a consequence, word frequencies and the coverage of senses are not balanced. The upshot of the one-year effort is the corpus of about 83,500 word occurrences, of which 58,596 are nouns, verbs, adjectives or adverbs. These open-class words are morphological variants of 7,595 distinct lemmas (5882 nouns, 747 verbs, 739 adjectives and 227 adverbs), and thereby calculated average number of samples per lemma is around 7.71. Remaining words are unambiguous entities, such as proper nouns, numbers, dates, conjunctions and punctuations. They are also included in the activity by assigning with specific synsets reserved for each of these categories. Table 4.3 lists distribution of words by grammatical categories.

Word Type	**Sample Size**	**Distinct Sample Size**
Noun	33,320	5,882
Verb	11,981	747
Adjective	9,591	739
Adverb	3,704	227
Number, Range, Date	4,460	
Punctuation	14,372	
Pronoun, Conjunction, etc.	6,046	
Total	83,474	7,595

Table 4.3: Distribution of word types.

In the inventory of TDK, how wide the range of words possible senses is quite irregular. While many of the members are monosemous, some of them may have up to 40-50 senses (see Figure 4.4). Hereby, needed time to annotate one target word diverges from a few seconds (for monosemous and homonyms) to several minutes (for subtle matters where even disagreements arise among taggers). In our corpus, 51,117 of the 58,596 open-class words have more than one sense choice. These polysemous occurrences have been central to the taggers effort during the annotation process. We list 10 most-frequent words with their rankings and number of candidate senses in Table 4.4.

Word	**Ranking**	**Number of Senses**
olmak (be, happen)	1,072	25
etmek (an auxiliary verb)	765	9
dolar (dollar)	622	1
hisse (share, stock)	616	3
bay (Mr.)	481	3
şirket (company, firm)	391	1
satmak (sell)	383	5
milyon (million)	380	2
yapmak (do, make)	375	20
demek (say, tell)	339	15

Table 4.4: 10 most-frequent words with their rankings and number of senses.

In the past Turkish WSD efforts, two distinct corpora have been employed: (1) METU-Sabanc Turkish Treebank and (2) Turkish Lexical Sample Dataset (TLSD). Several features of our resulting corpus are given in Table 4.5 in comparison with these sense tagged datasets.

Corpus	**Number of Samples**	**Distinct Lemmas**	**Word Coverage**	**Word Scope**	**Syntactic Parse**
METU	5,385	26	Lexical Sample	Sentence	Available
TLSD	3,616	35	Lexical Sample	Paragraph	Unavailable
Our Corpus	83,474	7,595	All-words	Paragraph	Available

Table 4.5: Comparison of Turkish sense annotated corpora.

Chapter 5

Features

To apply automatic methods to the problem of WSD, a set of features should be chosen to model the context. It is known that, supervised techniques are very sensitive to the selected features. Representing word occurrences can be challenging. To achieve a sufficient representation, distinctive features should be determined for different type of words. Also, it must be noted that effective features suggested for a specific language may not be eligible for the other ones.

Another factor that affects the disambiguation performance is the size of the context. It is very clear for us that a specific meaning a word assumes in context is directly related with the surrounding words. Therefore, the optimum extent of the window to the left and right of the head word should be established experimentally.

Contextual features usually come from the information arising from the aforementioned preprocessing tasks, such as morphological analysis and syntactic parsing. Morphological analysis supplies root form, major POS tag (e.g., verb, noun, etc.) and a set of morphological features (e.g., case markers, polarity, tenses, modality, etc.) for a single word. On the other hand, parsing gives valuable information about syntactically correlated parts (phrases) in a sentence. Based on these local and syntactic features, each word occurrence can be converted to a feature vector. For instance, Table 5.1 shows a simple example of possible feature vectors for the aligned sentence given in Figure 5.1.

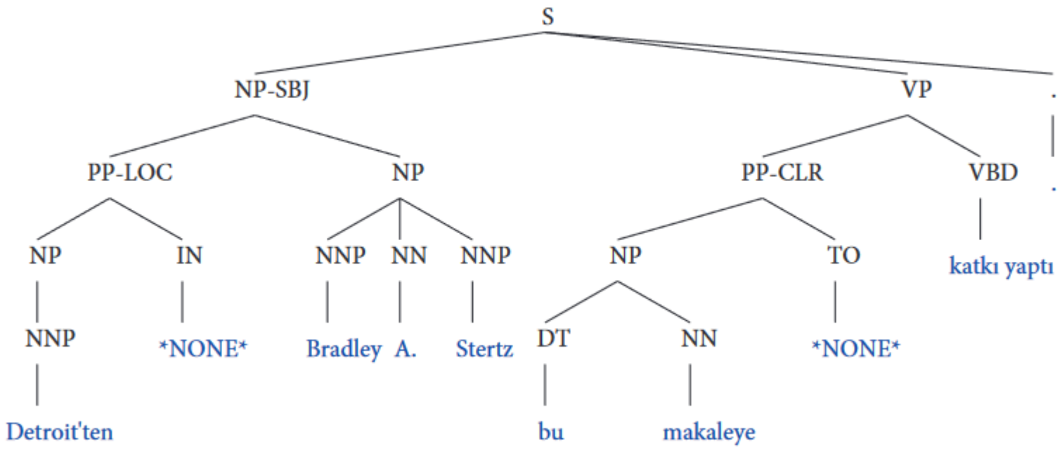


Figure 5.1: Turkish translated form of the Penn Treebank sentence: "Bradley A. Stertz in Detroit contributed to this article.".

In this example, we suppose that the target words are *makaleye* and *yaptı*. Our sample vectors include two local features (root form and major POS) and one syntactic feature (chunk tag) for the three words: head word itself, one word on the left (W_-1) and one word on the right (W_+1) of the head word.

bu	DET	NP	makale	NOUN	NP	katkı	NOUN	VP	$Synset_i$
katkı	NOUN	VP	yap	VERB	VP	NULL	NULL	NULL	$Synset_j$

Table 5.1: Example of feature vectors for the words *makaleye* and *yaptı*.

In this representation, NULLs refer to either inapplicable or unavailable cases. For example, there is no word on the right of *yaptı* (we disregard the punctuation), and as a consequence none of the attributes is applicable for W_+1. The last cell on the right is reserved for the sense label of words. $Synset_i$ refers to the assigned sense in the form of a synset number i, where i corresponds to the i-th synset in the dictionary.

There are several works on determining effective features for the disambiguation of Turkish words in a running text (see Section 3.2). According to the reported results, words that are closer to the head (located between $W_{-}2$ and $W_{+}2$) are more informative than the others. Also, fundamental features such as root words and POS tags is said to help to identify the intended meaning of words in context.

In this research, three distinct feature sets have been used to determine the effect of different kinds of attributes on the disambiguation performance. Most widely employed local features (surface forms, roots and the POS information in the window) are the starting point of our research. Following the initial experiments with this feature set, we have augmented our group of features with the employment of several morphological features reported in [23]. These features have been selected as the effective ones among the all available morphological features of the words in ±4 range. Table 5.2 and Table 5.3 list those effective features for nouns and verbs respectively.

Feature	**Position**
Noun + Acc	-2
Adj + PresPart	-2
Num	-1
Noun + Pnon	Head
Noun + Abl	Head
Noun + Ins	Head
Adj + PastPart	+1
Verb + Zero	+2

Table 5.2: Effective morphological features for nouns.

Feature	**Position**
Noun + Acc	-2
Adj + PresPart	-2
Num	-1
Noun + Pnon	Head
Noun + Abl	Head
Noun + Ins	Head
Adj + PastPart	+1
Verb + Zero	+2

Table 5.3: Effective morphological features for verbs.

Finally, we merge available syntactic features together with the local and morphological features to find out whether they contribute to the disambiguation score or not. In the parsed sentences, for each significant leaf node (e.g., word), we extract the first, second and third level of ancestors as the syntactic features. First ancestor is always a POS tag. On the other hand, upper ancestors give information about the chunks and relation between them. For ease of use, we symbolize the distinct feature sets used in the experiments as shown in Table 5.4.

Symbol	**Features**
FS_1	Surface form, root and POS
FS_2	Morphological features shown in the Tables 5.2 and 5.3
FS_3	Three syntactic features

Table 5.4: Feature set referrals.

Chapter 6

Evaluation

6.1 Evaluation Measures

In order to weigh up WSD systems, we use the following measures: precision (P), recall (R) and F1. These measures are usually interpreted in comparison with one or more baselines, which indicate an expected lower bound on the performance of automatic systems.

The assessment of WSD systems is usually performed in terms of precision and recall. The precision is defined as the number of correct predictions given by the system over the number of predictions given by the system.

$$P = \frac{\#\ correct\ predictions\ provided}{\#\ predictions\ provided} \tag{6.1}$$

It determines how successful are the answers given by the system. However, if the coverage is low, precision by itself does not say much about the performance. For example, simply classifying one instance successfully will yield 100% precision when the number of total predictions is exactly one.

Recall is defined as the number of correct predictions provided by the system over the total number of answers to be provided.

$$R = \frac{\#\ correct\ predictions\ provided}{\#\ total\ predictions\ to\ provide} \tag{6.2}$$

According to above formulas, $R \leq P$ and we have that $P = R$ when coverage is 100%. High recall (also referred to as accuracy), constitutes a more passable WSD measure than the precision.

Lastly, a measure which determines the weighted harmonic average of precision and recall, called the F1 measure, is defined as

$$F1 = \frac{2PR}{P + R} \tag{6.3}$$

F_1 measure is more useful than recall (accuracy), in comparison of systems with coverage lower than 100%.

6.2 Baselines

A baseline is a fixed reference point to which all other performances are compared. The random baseline and the most frequent sense (MFS) baseline are commonly employed baselines in the WSD literature.

In computation of the random baseline, a sense is assigned randomly from those available for each ambiguous word. On the other side, the MFS baseline consists in choosing the most frequent sense in an annotated corpus, independent of the words context. MFS baseline is also termed first sense baseline, since WordNet senses are ranked in accordance with the frequency of their occurrences in the SemCor corpus. However, this assumption may not be true for other sense inventories. Hence, a sense assignment made on the basis of the most frequent sense is more reliable than directly choosing first entry in a dictionary.

6.3 Experimental Results

During testing, we adopted K-fold cross-validation methodology. In this method, data is shuffled, and then divided into K equal parts to be used in K iterations.

On each iteration, one of those data units is selected for testing and remaining K-1 are used for training. In this way we assure that whole data is used for both training and testing. The overall accuracy of a system is calculated by taking the average of accuracies of the K folds. For this work, we have run our tests with the employment of 10-fold cross validation.

As we have discussed above, a limited number of words around the target word convey valuable tips about the meaning expressed by the word. In the evaluations, we have taken two words to the left and two words to the right together with the head word ($W_-2, W_-1, W_0, W_+1, W_+2$) into consideration to represent word usages in the form of feature vectors. For example, in the FS_1 representation, we use only three fundamental attributes, and therefore vector size becomes 16 (recall that one item in the vector holds the sense label).

We have applied various supervised methods (Naive Bayes, KNN, Rocchio, C4.5) to our all-words dataset to resolve disambiguation of all open-class polysemous entities. Classifiers have been tested with different feature groups to measure the effect of not only methods (algorithms) but also feature groups. First experimental setting is the employment of FS_1 (see Table 5.4). Table 6.1 presents the performance of each algorithm for both noun and verb sets when FS_1 employed separately. In this table, Random and MFS stand for the random baseline and most frequent baseline (refer to Section 6.2).

	Random	**MFS**	**NB**	**KNN**	**Rocchio**	**C4.5**
Noun	27.64	60.84	53.91	63.41	61.52	62.42
Verb	15.72	47.13	47.07	49.75	46.34	49.51
Total	24.04	58.75	52.84	61.25	59.17	60.41

Table 6.1: Accuracy (%) results for FS_1.

In the assessment of WSD systems, system accuracy is usually compared with the default strategy of selecting the most frequent sense in the training data. Despite its simplicity, MFS baseline is a real challenge for all-words WSD systems, as it is unlikely to acquire labeled data for the full lexicon. The results show that Naive Bayes method could not beat the MFS in both noun and verbs. The Rocchio algorithm also remains below the MFS for the verbs.

Although our corpus covers more than 50,000 polysemous word usages, average number of appearances per lemma is 7.71. As a result, classifiers suffer from data sparseness. Some of the words even appear only once in the corpus. In the lack of training data, test samples are inevitably predicted to belong to the first sense in the TDK dictionary.

In the second experiment, we have used the combination of FS_1 and FS_2 to be able to build more robust word experts. FS_2 includes the effective morphological features determined within the scope of a Turkish WSD research [23]. In that study, feature selection had promising results on a lexical sample dataset (TLSD). This dataset contains minimum 100 samples for each ambiguous word. We want to see whether we can exceed the success rate of the previous experiments with this augmented feature set. Table 6.2 reports the performance of the systems for the setting of $FS_1 + FS_2$.

	Random	**MFS**	**NB**	**KNN**	**Rocchio**	**C4.5**
Noun	27.64	60.84	59.04	63.68	62.24	62.42
Verb	15.72	47.13	50.00	50.00	47.80	49.51
Total	24.04	58.75	57.63	61.54	59.99	60.41

Table 6.2: Accuracy (%) results for $FS_1 + FS_2$.

The outcomes show that adding morphological features to the basic feature set brings a little improvement from the previous results. This could be due to the fact that those effective feature sets have been determined on a different corpus that includes only the samples of a limited number of ambiguous words. Actually, words usually exhibit specific behavior, suggesting that the best results could be achieved with specific features sets for each word. For this case, best system increase is observed for the Nave Bayes with a + 5.13 for nouns and + 2.93 for verbs. Additionally, it is interesting to note that, outcomes of the C4.5 are same with the previous experiment.

In the final step, we conduct experiments by merging the three feature sets together ($FS_1 + FS_2 + FS_3$). In this configuration, we aim to analyze the contribution of syntactic features. Extraction of this kind of knowledge requires high effort in all ways, either starting from scratch or aligning from a treebank in different language. So, we want to know if they are helpful for WSD. Results of joint setting of whole feature sets are shown in Table 6.3.

	Random	**MFS**	**NB**	**KNN**	**Rocchio**	**C4.5**
Noun	27.64	60.84	59.72	65.71	64.90	63.23
Verb	15.72	47.13	50.74	50.50	51.23	49.51
Total	24.04	58.75	58.32	63.34	62.77	61.09

Table 6.3: Accuracy (%) results for $FS_1 + FS_2 + FS_3$.

Results show that using a richer feature set (including the syntactic ones) improves the disambiguation performance. Without exception, all the algorithms yielded their best scores under the setting of ($FS_1 + FS_2 + FS_3$). In the total performance of the algorithms, we have observed an improvement ranged between 0.68% (C4.5) and 2.78% (Rocchio).

To our knowledge, this is the first time the information of syntactic phrases is being used in a Turkish WSD study. In some earlier studies [19], METU-Sabanci

Turkish Treebank [20] has been exploited, as it provides syntactic features. However, it is a dependency treebank that presents the information of how words relate to each other, rather than the skeletal structure of a sentence. Considering the scores of this experiment, we can say that the treebanks that annotate the phrase structure make possible to acquire valid source of knowledge to be integrated in a WSD system. Table 6.4 summarizes the accuracy results of the experiments in comparison with each other.

	Random	**MFS**	**NB**	**KNN**	**Rocchio**	**C4.5**
FS_1	24.04	58.75	52.84	61.25	59.17	60.41
$FS_1 + FS_2$	24.04	58.75	57.63	61.54	59.99	60.41
$FS_1+FS_2+FS_3$	24.04	58.75	58.32	63.34	62.77	61.09

Table 6.4: Comparison results of total accuracy (%) for distinct experimental settings.

Chapter 7

Conclusion

In the first part of this thesis, we reported our experience on hand tagging of TDK senses in a Turkish-English parallel treebank, aligned from Penn Treebank. This corpus has already been parsed, and enhanced with morphological features before the semantic annotation process, presented here. From initial translation endeavors to the current sense labeling degree, common text files have been processed in a progressive manner. That is, each distinct study made full use of the previous one without corrupting it and produced a cumulative data set. Such a technique of exploiting resources at hand, either a treebank in different language or morphological analyses in native language, proved to be a useful method that accelerates the corpus construction process.

Turkish is one of the languages that need much more linguistic resources to speed up NLP research. Creation of this dataset will contribute to this call, offering an all-words sense annotated corpus. We hope that this corpus will also be a useful resource for various NLP studies, since it is a full-coverage material that provides syntactic parse and morphological analysis together with sense annotations.

The resulting corpus is superior to the other annotated corpora of Turkish, in several aspects. First, word coverage is much better, since all words are labeled. This enables to make all- words WSD experiments. Second, root forms and morphological structure are already on hand. This eliminates the need for the external tools, such as morphological analyzer. Also, syntactic features available in the parsed sentences make possible to acquire more information about the words.

In the second part, we have implemented several supervised classifiers (Naive Bayes, KNN, Rocchio, C4.5) to carry out all-words WSD experiments on the corpus. During the tests, we have used three different feature sets: (FS_1) fundamental local features, (FS_2) effective morphological features and (FS_3) syntactic features. Throughout the evaluations, selecting first sense in the absence of training data has been applied as a back-off strategy.

In the first experiment, we have run automatic methods with the fundamental local features. This experiment provides insight into the success level of a simple Turkish WSD system in a real-like setting. Systems have tried to disambiguate more than 50,000 occurrences of a 7,595 distinct word, unlike that of the lexical sample tasks which take the usages of a small number of pre-determined words as input. Then, we have utilized local features and morphological features ($FS_1 + FS_2$) together for the second experiment. According to the results, all systems yielded better scores than the previous one. However, it is important to note that the progress in the disambiguation rate is too little in comparison to the lexical sample task [23] for which those effective features provided considerable increase in the success rate. Finally, we have applied the joint setting of all feature groups ($FS_1 + FS_2 + FS_3$) to measure additional improvements in the results. As a result, an improvement ranged between 0.68% and 2.78% in accuracy is observed. This experiment shows that syntactic features found in the skeletal structure of a sentence can increase the accuracy when training data is scarce.

The all-words WSD is an extensive and complicated task. From the overall results, we can say that making a qualitative jump in the system performances is a big challenge. To achieve higher performance levels, following issues have to be kept in mind:

(1) We need a coarse-grained Turkish sense inventory. Sense divisions in the TDK dictionary are often too fine-grained for WSD applications. If humans cannot

agree on the intended meaning of a word, what does it mean if an automatic system predicts the sense of a polysemous word?

(2) Sometimes the meaning of a word cannot be understood with only by the sentence it appears. In such cases, missing information should be extracted from the preceding or succeeding context. This indicates the need for the larger context around the target word. Word occurrences in paragraph scope allows to obtain extra information compared to the sentence scope.

(3) Existing Turkish hand-tagged corpora, either lexical sample or all-words, do not seem enough for sufficient WSD systems. Therefore, knowledge acquisition still remains as an important issue for Turkish WSD researches.

(4) The feature sets used to model the context directly affect the performance of ML techniques. Thus, effective feature sets should be explored for sufficient representation of Turkish language. Aggregation of different types of features could be a good approach, but it requires the complex analysis of text, such as syntactic parse and morphological analysis. The experiments performed in this study showed that morphological, local and syntactic features include complementary information.

(5) In the WSD literature, different classification algorithms have been superior to other approaches in different tasks. So far, nobody has claimed that any algorithm is the best for the problem of WSD. In the test phase, different approaches should be employed to get the optimum option.

(6) Knowledge-based methods can be a better choice than the first sense in the lack of training data.

References

[1] O. T. Yildiz, E. Solak, O. Görgün, and R. Ehsani, "Constructing a Turkish-English parallel treebank," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014, pp. 112–117.

[2] M. P. Marcus and M. A. Marcinkiewicz, "Building a large annotated corpus of english: The penn treebank," *Computational Linguistics*, vol. 19, pp. 313–330, 1993.

[3] O. Görgün, O. T. Yildiz, E. Solak, and R. Ehsani, "English-Turkish parallel treebank with morphological annotations and its use in tree-based SMT," in *International Conference on Pattern Recognition and Methods (ICPRAM)*, 2016, pp. 510–516.

[4] S. Banerjee and T. Pedersen, "Extended gloss overlaps as a measure of semantic relatedness," in *Proceedings of the 18th international joint conference on Artificial intelligence*, 2003, pp. 510–516.

[5] P. Basile, A. Caputo, and G. Semeraro, "An enhanced Lesk word sense disambiguation algorithm through a distributional semantic model," in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*, 2014, pp. 1591–1600.

[6] E. Agirre and D. Martinez, "Learning class-to-class selectional preferences," in *Proceedings of the 5th Conference on Computational Natural Language Learning*, 2001, pp. 1–8.

[7] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller, "Wordnet: An online lexical database," *International Journal of Lexicography*, vol. 3, pp. 235–244, 1990.

[8] R. Navigli and S. P. Ponzetto, "Knowledge-rich word sense disambiguation rivaling supervised systems," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2010, pp. 1522–1531.

[9] R. Tripodi and M. A. Pelillo, "A game-theoretic approach to word sense disambiguation," *Computational Linguistics*, vol. 43, pp. 31–70, 2017.

[10] R. Mihalcea, "Co-training and self-training for word sense disambiguation," in *Proceedings of the 8th Conference on Computational Natural Language Learning*, 2004, pp. 33–40.

[11] E. Agirre, O. Ansa, D. Martinez, and E. Andhovy, "Enriching WordNet concepts with topic signatures," in *Proceedings of the NAACL Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*, 2001, pp. 23–28.

[12] R. Mihalcea, "Word sense disambiguation with pattern learning and automatic feature selection," *Natural Language Engineering*, vol. 8, pp. 343–358, 2002.

[13] G. A. Miller, C. Leacock, R. Tengi, and R. T. Bunker, "A semantic concordance," in *Proceedings of the ARPA Workshop on Human Language Technology*, 1993, pp. 303–308.

[14] R. Mihalcea, "Bootstrapping large sense tagged corpora," in *Proceedings of the 3rd International Conference on Languages Resources and Evaluations*, 2002.

[15] A. Gliozzo, B. Magnini, and C. Strapparava, "Unsupervised domain relevance estimation for word sense disambiguation," in *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004, pp. 380–387.

[16] J. Clear, "The British National Corpus," 1993.

[17] Y. S. Chan, H. T. Ng, and Z. Zhong, "NUS-PT: Exploiting parallel texts for word sense disambiguation in the English all-words tasks," in *Proceedings of the 4th International Workshop on Semantic Evaluations*, 2007, pp. 253–256.

[18] H. T. Ng and H. B. Lee, "Integrating multiple knowledge sources to disambiguate word senses: An examplar-based approach," in *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, 1996, pp. 40–47.

[19] Z. Orhan, E. Çelik, and N. Demirgüç, "SemEval-2007 task 12: Turkish lexical sample task," in *Proceedings of the 4th International Workshop on Semantic Evaluations*, 2007, pp. 59–63.

[20] K. Oflazer, B. Say, and N. B. Atalay, "The Annotation Process in the Turkish Treebank," in *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora*, 2003.

[21] B. İlgen, E. Adalı, and A. C. Tantuğ, "Exploring feature sets for turkish word sense disambiguation," *Turkish Journal of Electrical Engineering Computer Sciences*, vol. 24, pp. 4391–4405, 2016.

[22] B. İlgen, E. Adalı, and A. C. Tantug, "Building up lexical sample dataset for turkish word sense disambiguation," in *IEEE International Symposium on Innovations in Intelligent Systems and Applications*, 2012, pp. 1–5.

[23] B. İlgen, E. Adalı, and A. C. Tantuğ, "The impact of collocational features in turkish word sense disambiguation," in *IEEE 16th International Conference on Intelligent Engineering Systems*, 2012, pp. 527–530.

[24] S. Stamou, K. Oflazer, K. Pala, D. Christodoulakis, D. Cristea, D. Tufis, S. Koeva, G. Totkov, D. Dutoit, and M. Grigoriadou, "BalkaNet: A multilingual Semantic Network for Balkan Languages." in *Proceedings of the First International WordNet Conference*, 2002.