WORD SENSE DISAMBIGUATION, NAMED ENTITY RECOGNITION, AND SHALLOW PARSING TASKS FOR TURKISH

OZAN TOPSAKAL

IŞIK UNIVERSITY
2019

WORD SENSE DISAMBIGUATION, NAMED ENTITY RECOGNITION, AND SHALLOW PARSING TASKS FOR TURKISH

OZAN TOPSAKAL

B.S., Computer Engineering, IŞIK UNIVERSITY, 2013

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

WORD SENSE DISAMBIGUATION, NAMED ENTITY RECOGNITION, AND SHALLOW PARSING TASKS FOR TURKISH

OZAN TOPSAKAL

APPROVED BY:

Prof. OLCAY TANER YILDIZ Işık University
(Thesis Supervisor)

Prof. Dr. M. Oğuzhan KÜLEKÇİ İstanbul T. University

Dr. Nilgün Güler BAYAZIT Yıldız T. University

APPROVAL DATE: 04 / 02 / 2013

WORD SENSE DISAMBIGUATION, NAMED ENTITY RECOGNITION, AND SHALLOW PARSING TASKS FOR TURKISH

Abstract

People interactions are based on sentences. The process of understanding sentences is thru converging, parsing the words and making sense of words. The ultimate goal of Natural Language Processing is to understand the meaning of sentences. There are three main areas that are the topics of this thesis, namely, Named Entity Recognition, Shallow Parsing, and Word Sense Disambiguation.

The Natural Language Processing algorithms that learn entities, like person, location, time etc. are called Named Entity Recognition algorithms.

Parsing sentences is one of the biggest challenges in Natural Language Processing. Since time efficiency and accuracy are inversely proportional with each other, one of the best ideas is to use shallow parsing algorithms to deal with this challenge.

Many of words have more than one meaning. Recognizing the correct meaning that is used in a sentence is a difficult problem. In Word Sense Disambiguation literature there are lots of algorithms that can help to solve this problem.

This thesis tries to find solutions to these three challenges by applying machine learning trained algorithms. Experiments are done on a dataset, containing 9,557 sentences.

Keywords: Natural Language Processing, NLP, Named Entity Recognition, NER, Shallow Parsing, Word Sense Disambiguation, Machine Learning

TÜRKÇE İÇİN KELİME ANLAMLANDIRMA, ADLANDIRILMIŞ VARLIK TANIMA VE SIĞ AYRIŞTIRMA

Özet

İnsanların birbiriyle diyalogları cümlelerle olmaktadır. Cümlenin anlaşılması, kelimelere yakınsayarak, onları ayrıştırarak ve cümle içerisinde kullanılan ideal anlamlarını bularak olur. Doğal Dil İşleme'nin nihai amacı cümleyi anlamaktır. Bu tezin konusu üç alandan oluşmaktadır: Adlandırılmış Varlık Tanıma, Sığ ayrıştırma ve Kelime Anlamlandırma'dır.

"İnsan", "yer", "zaman" gibi varlıkları öğrenebilen Doğal Dil Geliştirme algoritmalarına Adlandırılmış Varlık Algoritmaları denir.

Cümleleri ayrıştırma Doğal Dil İşleme'nin en büyük meydan okumalarından birisidir. Zaman ve doğruluğu arttırma ters orantılı olduğundan dolayı Sığ Ayrıştırma algoritmaları bu konudaki en iyi çözümlerden biridir.

Bir çok kelimenin birden çok anlamı vardır. Cümle içinde kullanılan kelimenin doğru anlamını alglamak zorlu bir problemdir. Kelime Anlamlandırma literatüründe bu problemi çözümlemek için bir çok algoritma mevcuttur.

Bu tezde bu üç alan için makine öğrenimi algoritmalarıyla çözümler üretilmeye çalışılmıştır. Deneyler 9,557 cümlelik bir veri kümesi üzerinde yapılmıştır.

Anahtar kelimeler: Doğal Dil İşleme, Adlandırılmış Varlık Tanıma, Sığ Ayrıştırma, Kelime Anlamlandırma, Makine Öğrenmesi

Acknowledgements

This study was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) Grant No: 116E104

To my family who supported me throughout my life...

Table of Contents

List of Tables

List of Figures

List of Abbreviations

NLP	**N**amed **E**ntity **R**ecognition
WSD	**W**ord **S**ense **D**isambiguation
NER	**N**amed **E**ntity **R**ecognition
SP	**S**hallow **P**arse
RF	**R**andom **F**orest
KNN	**K** - **N**earest **N**eighbour
NB	**N**aive **B**ayes

Chapter 1

Introduction

All social species, from ants to dolphins and to monkeys in the world, communicate with each other. Only the humankind has achieved to develop a systematic communication system, that is more advanced than simple gestures, called language. With the development of society many different and unique languages, and from them even language families have emerged. From those, some of the languages are analytic like English, and some of them are agglutinative like Turkish.

With the development of computers, in the early 1950s, Natural Language Processing scientists had begun the research about artificial intelligence and, its subdisciplines which try to improve the communication between humans and computers. The main objective of NLP is to make computers understand the inscribed and verbal statements of human, so to improve human-computer interaction. NLP gathers some of the most sophisticated fields of computer science to achieve this goal like Computational linguistics, machine learning, and programming. Today NLP is one of the most popular fields of computer science.

There are six levels of language interpretation that NLP systems work, which can be listed as “discourse level“, “lexical level“, “morphological level“, “semantic level“, “syntactic level“, and “programmatic level“. The brief explanations of them are as follows:

Discourse level works on how sentences are related to each other. The lexical level works on how the word parts (morphemes) combinations make or change the word. The morphological level works on what morpheme is and the role they play in making a word. The semantic level works on the meaning of words and what, who, how or when they refer to.

The syntactic level works on the syntax of sentences. The pragmatic level works on the meaning of a word that makes the most sense in the current sentence.

These levels have many working areas. This thesis is about three of those areas, which are Named Entity Recognition Shallow Parsing, and Word Sense Disambiguation.

The first area, namely Named Entity Recognition is one of the semantic level subtasks, which aims to recognize the words having predefined classes like “person“, “institution“, “place“, “time expression“, or “currency“ etc. It is important to extract these classes of the words in a sentence in order to understand the syntax of it.

The second area, this paper will cover is Shallow Parsing, which is also a semantic subtask. Parsing in NLP is a difficult challenge. Shallow parsing has emerged from the absence of tools that can give reasonable information in a short notice of time. Shallow parsing focuses on the identification of the word as subject, object or predicate.

The third area of the semantic level this paper is going to discuss is Word Sense Disambiguation. There are many words which have more than one meaning. This area is interested in selecting the best meaning suitable for a word according to the context of the sentence.

In NLP, there are a lot of research articles on analytic languages. However, researches about agglutinative languages are limited. This thesis is about one of these agglutinated, Turkish.

In this thesis, we apply six different machine learning algorithms to NER, Shallow Parsing, and Word Sense Disambiguation problems. In order to do this, a dataset, which has 9,557 sentences and more than 65,000 words, is generated.

This thesis expands the works of on Ertopçu et.al [1], Topsakal et.al [2], Açıkgöz et.al [3], and Yıldız et. al - 2018 [4]. The dataset in the previous papers [1], [2], [3] include 1,400 sentences and 13,194 words. The dataset in the paper [4] contains 4,400 sentences and approximately 40,000 words. The corpus of this thesis has 9,557 sentences, and 69,711 words which are annotated for NER, SP, and WSD problems.

[1], [2], [3] uses Random Forest ([1], [3]), Multilayer Perceptron ([1], [2], [3]), Linear Perceptron ([1], [2]), K-Nearest Neighbor ([2], [3]), Naive Bayes ([2]), Rocchio (in [3]), and C45 ([3]) classifiers. In this thesis, we use Dummy, Naive Bayes, Rocchio, C45, K-Nearest Neighbor, Random Forest classifiers.

In this thesis for NER problem we also added “person“, “location“, “organization“ gazetteers and also a new feature namely “predicate“.

In previous works [1], [2], [3] window size is variable, whereas in this thesis window size is fixed to two.

In this thesis, parameter tuning is applied for all classifiers. Afterward, feature selection is applied to get final results.

This thesis is organized as follows. We define NER, Shallow Parsing, and Word Sense Disambiguation problems in Chapter 2. We give previous works in Chapter 3. In Chapter 4 we explain the data. In Chapter 5 we introduce all the algorithms that are used. We presented the features that are used in algorithms in Chapter 6. In Chapter 7 we showed the experiments and results for parameter tuning and feature selection. Finally in Chapter 8 explain the conclusion and future works.

Chapter 2

Problems

2.1 Named Entity Recognition

NER is a sub-task of semantic level language interpretation which extracts entity information of a word. In a sentence, a word that denotes proper names like "person", "location", or "organization", and "date", "time", or "money" is called named entity. Here is an example text with marked named entities:

"[$_{ORG}$ Bursa Deniz Otobüsleri] [$_{DATE}$ 23 Nisan] sebebiyle [$_{LOC}$ Bursa] [$_{LOC}$ Yenikapı] arası tarifelerini [$_{MONEY}$ 40 TL'ye] sabitlediğini duyurdu."

"[$_{ORG}$ Bursa Sea Ways] announced that [$_{LOC}$ Bursa] [$_{LOC}$ Yenikapı] sailing prices had fixed to [$_{MONEY}$ 40 TL] because of [$_{DATE}$ April 23th]."

This sentence contains five named entities including three words labeled as ORGANIZATION, two words labeled as LOCATION, two words labeled as DATE, and one word labeled as MONEY. Table 2.1 shows typical generic named entity types.

In general two types of problems are encountered, which are to classify the word whether or not a named entity and the words may be used of with either a date, a person, a location, or an organization tag. For example, *Beşiktaş* may be used as a location, or football club (an organization) *Beşiktaş*.

Table 2.1: List of name entity types and examples

Tag	Sample Categories	Example
PERSON	people, characters	Yaşa **Mustafa Kemal Paşa** yaşa! Viva **Mustafa Kemal Pasha** Viva!
ORGANIZATION	corporations, teams	**İş Bankası** 1924 yılında kuruldu. **İş Bankası** was founded in 1924.
LOCATION	regions, place, city	Ülkemizin başkenti **Ankara'dır**. **Ankara** is the capital of our country.
TIME	time, date	Uçak **çarşamba** saat **18:00'da** kalkacak. The plane will depart on **Wednesday** at **18:00**.
MONEY	monetarial expressions	**1 dolar** bugün itibariyle **5.42 tl'den** işlem görüyor. **1 dollar** traded at **5.42tl** as of today.

First, we tag all words with certain named entity labels depending on the context of those words. After tagging the training data, set of features are selected for discerning the distinct named entities per input word. Table 2.2 shows the example sentence represented with tag labels and three possible features, namely the root form of the word, the part of speech (POS) tag of the word, and a boolean feature for checking the capital case.

Given such training data, new sentences can be labelled by a classifier like a neural network or a decision tree. Figure 2.1 shows a sample operation of a classifier that kind at the point where the word *Yenikapı* is going to be labeled. For this classifier, the window size is two. The assumption is that a context window includes two words before and two words after.

Table 2.2: Classification problem - Named entity recognition

Word	Features				Label
	Root	Pos	Capital	...	
Bursa	Bursa	Noun	True	...	ORGANIZATION
Deniz	Deniz	Noun	True	...	ORGANIZATION
Otobüsleri	Otobüs	Noun	True	...	ORGANIZATION
23	23	Number	False	...	DATE
Nisan	Nisan	Noun	True	...	DATE
sebebiyle	sebep	Conjunction	False	...	NONE
Bursa	Bursa	Noun	True	...	LOCATION
,	,	Punctuation	False	...	NONE
Yenikapı	Yenikapı	Noun	True	...	LOCATION
arası	ara	Adverb	False	...	NONE
tarifelerini	tarife	Noun	False	...	NONE
40	40	Number	False	...	MONEY
TL'ye	TL	Noun	True	...	MONEY
sabitlediğini	sabit	Noun	False	...	NONE
duyurdu	duy	Verb	False	...	NONE
.	.	Punctuation	False	...	NONE

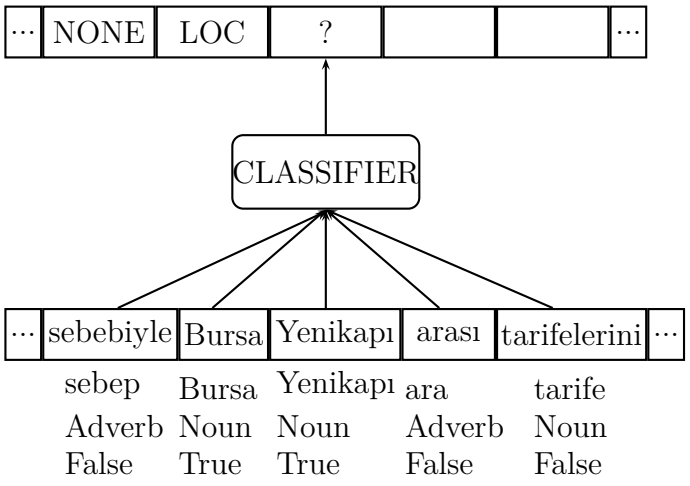

Figure 2.1: A named entity recognition classifier approach. At the moment, it can be seen that the classifier is on Yenikapı to label. Features are provided from the sentence, words, POS tags are involved

2.2 Shallow Parse

A computer has to understand not only the meaning but also the hierarchical sequence of words in order to learn the syntax of the sentences, except that is a very challenging problem in NLP. Because some of the tools give too much information, and they are slow. However, most of the tasks dont require complex parse trees. On the other hand, some tools are fast, but they have insufficient information. Shallow Parse can provide enough information in a reasonable time. Shallow parse is an identifier of sentence elements. These elements are *Özne* "Subject", *Nesne* "Direct Object", *Yüklem* "Predicate", *Zarf Tümleci* "adverbal clause", and *Dolaylı Tümleç* "indirect object". Simple bracket notation is enough to figure where and what kind of shallow parse chunks in the text because when text is parsed, it does not contain hierarchic structure. The following sample context shows marked shallow parse chunks:

[$_{OZNE}$ Ḃursa Deniz Otobüsleri] [$_{ZARF\ TUMLEC}$ 23 Nisan sebebiyle] [$_{NESNE}$ yarınki seferlerini] [$_{YUKLEM}$ arttırmış].

[$_{SUBJECT}$ Ḃursa Deniz Otobüsleri] [$_{PREDICATE}$ increased] [$_{DIRECT\ OBJECT}$ their flights of tomorrow] [$_{ADVERBIAL\ CLAUSE}$ due to 23 April].

This sentence contains 5 shallow parse chunks including 3 words labeled as ÖZNE (SUBJECT), 3 words labeled as ZARF TÜMLECİ (ADVERBIAL CLAUSE), 2 words labeled as NESNE (DIRECT OBJECT), and 1 word labeled as YÜKLEM (PREDICATE). Table 2.3 shows sample shallow parse tags and the tag identifier questions.

In shallow parsing, the objective is to find the connections and place of the word in the sentence and to identify its classification. Literal grouping which is one of the common approaches for shallow parsing proceeds in two steps: First training the classifier, and second selecting a group of features for each word input. Training classifier is tagging words from particular chunks for labeling. Feature selection is for segregating different chunks. Table 2.4 shows the sample text represented

Table 2.3: List of shallow parse chunk tags

Tag	Question	Example
ÖZNE	Who	İlk cumhurbaşkanı **Atatürk** oldu. **Atatürk** was the first president of the republic.
ZARF TÜMLECİ	When	**Yarın** işe başlıyor. She starts work **tomorrow**.
DOLAYLI TÜMLEÇ	Where	**Karanlık bir sokakta** yürüyordu. She was walking on **a dark street**.
NESNE	What	Gelirken **ekmek** al. Buy **bread** when you come.
YÜKLEM	Predicate	Paramız **değerlenmiyor**. Our money is not **valued**.

with chunk labels and three possible features, namely the root of the word, POS tag, and a boolean feature for checking the capital case.

Table 2.4: Shallow parse classifier

Word	Features				Label
	Root	Pos	Capital	...	
Bursa	Bursa	Noun	True	...	ÖZNE
Deniz	deniz	Noun	True	...	ÖZNE
Otobüsleri	otobüs	Noun	True	...	ÖZNE
23	23	Number	True	...	ZARF TÜMLECİ
Nisan	Nisan	Noun	False	...	ZARF TÜMLECİ
sebebiyle	yeni	Conjunction	False	...	ZARF TÜMLECİ
yarınki	yarın	Noun	False	...	NESNE
seferlerini	sefer	Noun	False	...	NESNE
arttırmış	art	Verb	False	...	YÜKLEM
.	.	Punctuation	False	...	HİÇBİRİ

Given such training data, new sentences can be labeled by a classifier like a neural network or a decision tree. Figure 2.2 shows a sample operation of a classifier that kind at the point where the word *Nisan* is going to be labeled. For this classifier, the window size is 2. The assumptions are that a context window includes two words before and two words after.

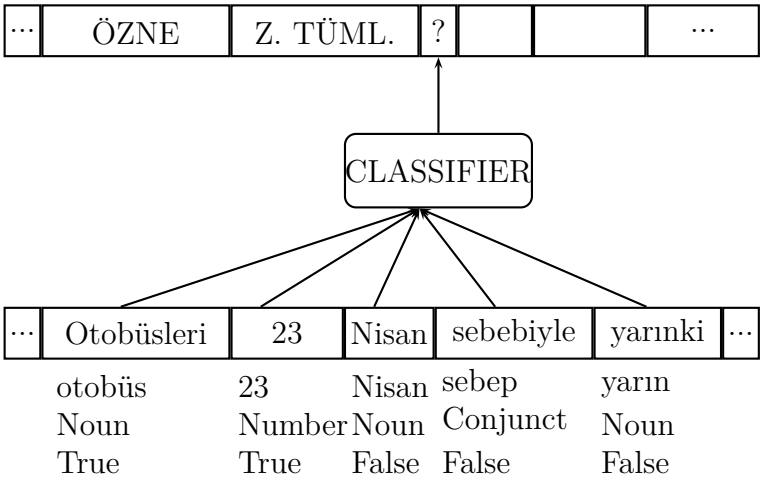

Figure 2.2: Shallow parsing classifier based approach. The classifier slides through sentence, parsing words with context window. At the moment parser is trying to label *Nisan*. Features are provided from the text are words, POS tags etc.

2.3 Word Sense Disambiguation

Word Sense Disambiguation is a branch of the semantic sub-tasks, which tries to choose the correct meaning of a word. WSD algorithms take the word from a string and the pre-defined means list as parameters to define closest to right meaning. Moreover, in isolated studies dictionary, or thesaurus like WordNet is also added to the parameters. Table 2.5 represents a sample word *dil* and the sample meanings, which are referred to noun "tongue", to the verb "slice" or another noun "language".

Table 2.5: Some definitions and examples for the sense tags for 'dil'

Sense	Definition	Example
dil[1]	Konuşma organı	**Dil** en güçlü kastır.
	tongue	**Toungue** is the strongest muscle.
dil[2]	Bir bütünü ince ve yassı parçalara ayırarak kesmek.	Salamı ince ince **dildim.**
	Slicing	I **sliced** the salami.
dil[3]	Lisan	Yabancı **dilini** geliştir.
	Language	Improve your foreign **language**.

There are two types of WSD tasks in the literature which are **lexical sample**, and **all-words**. **lexical sample** is an induction like task, which process is thru selection of a small number of words along with their group of senses, then each correct sense is manually matched with each word.

On the other hand **all-words** is a deduction like task, which gets whole sentences with containing words sentences. All the words are classified by annotators. Classifiers choose the best accurate sense among sets. After the words get the sense labels, for each word, a group of features is selected on the purpose of discrimination. Table 2.6 shows the sample text represented with sense labels and three possible features, namely the root form of the word, the part of speech (POS) tag of the word, and a boolean feature for checking the capital case.

Table 2.6: All-words WSD as a classification problem

Word	Features				Label
	Root	Pos	Capital	...	
Onun	o	Pronoun	True	...	o^{1}
yüzünden	yüz	Noun	False	...	yüz^{3}
yüz	yüz	number	False	...	yüz^{2}
yerde	yer	noun	False	...	yer^{1}
yüzüm	yüz	noun	False	...	yüz^{1}
kızarıyor	kızar	Verb	False	...	kızar^{3}
.	.	Punctuation	False	...	.^{1}

Given such a training data, a classifier like a neural network or decision tree can be trained for labeling new words. Figure 2.3 shows the operation of a classifier that, points where the word *yüz* is labeling next at. For this classifier, the window size is two. The assumptions is that a context window includes two words before and two words after.

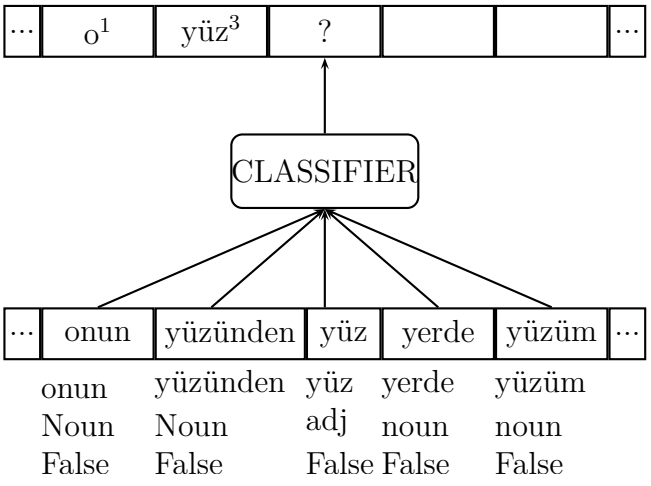

Figure 2.3: Word Sense Disambiguation: For all-words Classifier based approach. The classifier slides through sentence, labelling words with context window. At the moment labeler is trying to label *yüz*. Features are provided from the text are words, POS tags etc.

Chapter 3

Previous / Related Works

3.1 Named Entity Recognition

3.1.1 Linguistic Background

While NER is a rather unproblematic task among NLP studies in well-studied languages like English, it faces certain challenges when dealing with a language like Turkish. One of the central foci of literature on Turkish linguistics has been the complexity of Turkish morphology. Turkish is a textbook example of an agglutinative language, i.e. words in their surface form may contain various morphemes, especially suffixes. The main problem posed by such languages is lexical sparsity [5], which, in general, can be considered as a challenge for earlier steps such as the Morphological Analysis and Disambiguation tasks which feed NER.

Not only the morphology but also the syntactic structure of Turkish is a fertile source for challenging issues. Turkish is typically treated as a subject-object-verb (SOV) language yet it has a rather free word order. In other words, the constituents of a sentence can occur in any order with slight modifications in the sentential meaning. A particular order is chosen mainly on pragmatic grounds [6]. Therefore, the position of a word within a sentence does not provide any clues about whether it is a Named Entity or not.

As an additional problem specific to NER, there exist many proper nouns in Turkish which are derived from common names (through suffixation such as the following names of cities in Turkey: *Denizli* derived from *deniz* "sea", compounding such as *Pamukkale* from *pamuk+kale* "cotton+fort" or zero-derivation such as *Tokat* from *tokat* "slap"). In the speech, prosodic cues are helpful in distinguishing proper nouns, especially place names, from common nouns due to their idiosyncratic stress pattern [7]. Yet there is no orthographic correlates to stress. Instead, informal texts at least, orthographic clues can be helpful in distinguishing a proper noun, which starts with a capital letter, from a common noun. In sentence-initial position, however, all words begin with a capital letter and hence this clue is not available.

In short, several linguistic features of Turkish, such as its rich morphology, free word-order, as well as derivation as a word-formation process frequently employed in forming proper names yield problems for NER tasks.

3.1.2 Computational Background

Study on NER has big attention in the literature. However, most of the models are specific to the language of focus. A language-independent method is proposed in [8] which is a bootstrapping algorithm based on iterative learning. The method relies on word interval and contextual clues. This work is tested on Turkish, Romanian, English, Greek and Hindi and can be considered as the first work on Turkish NER.

Turkish-specific studies are rather narrow and rare compared to the other worldwide speaking languages. [9] was the first known Turkish NER research, which was an information extraction based work. [10] which was also an early study that follows was focused on conditional random fields that used morphological and lexical properties. These were built the infrastructure of nowadays leading projects, however, the dataset used is gathered from real natural language data. Taking Turkish NER studies into account, models based on Hidden Markov Models

(HMMs) [9], Conditional Random Fields (CRFs) [11] and rule-based [12] studies are presented on data gathered from news reports. Formal texts like news are written on certain rules of language. Authors of such texts follow the grammatical and orthographical rules of the language in question and thus generate statistically less volatile data. Social media like Twitter is also providing data to NLP studies, however unlike formal text, social media texts neither have the obligation to follow spelling rules, nor words need to be complete. Furthermore, this kind of texts may have emotions, abbreviated words or words that used in a different sense than usual. An experimental Twitter text [13] work reveals the difference between social media and formal text.

3.2 Shallow Parse

3.2.1 Linguistic Background

In linguistics, sentences are assumed to be constructed from a rule-based combination of several constituents. In other words, sentences are not considered as linear strings of words - rather, they are treated as hierarchically organized structures consisting of phrases (such as NPs, i.e. noun phrases, and VPs, i.e. verb phrases) which may, in turn, contain other phrases. Determining the constituents of a sentence is a renowned problem in syntax. Various constituency tests have been proposed in the linguistics literature, yet they are not always reliable or may lack a cross-linguistic applicability [14].

At the very basics, a sentence consists of a subject NP and a predicate VP. The predicate, in turn, may contain a variety of phrases with several grammatical roles, such as the direct or indirect object, or an oblique object. Turkish is typically treated as a head-final subject-object-verb (SOV) language, yet a central problem with Turkish is that it does not have a strict word order. In other words, the constituents of a sentence can be scrambled such that they can occur in any order

with slight modifications in the sentential meaning. A particular order is chosen mainly on pragmatic grounds [6].

In general, a language in its written form gives us orthographic clues about word boundaries but not about the phrase or constituent boundaries (except for punctuation marks in some cases). In a speech, prosodic cues such as stress and intonation patterns may help the listener in parsing, yet such cues are unavailable in written texts. In Turkish, the problem of parsing is even more advanced, as not only content words in their base forms but also functional morphemes, such as affixes, may be the source of ambiguity which cannot be resolved without the context. One of the central foci of literature on Turkish linguistics has been the complexity of Turkish morphology. Turkish is a textbook example of an agglutinative language, i.e. words in their surface form may contain various morphemes, especially suffixes. While certain affixes have clear functions/meanings, there exist others for which the meaning can only be determined within a discourse. Derivational affixes may change the POS of a word, which makes it possible for root to serve grammatical roles that are not typical for its root's POS-tag. Even without any derivational process, the POS distinction among the nominal words (especially between nouns and adjectives) is far from being absolute. For instance, an adjective can be used as a noun and hence can occupy the subject position (the head of NP) within a sentence.

Tok / Aç-ın / hal-in-i /anla-maz.

full/ hungry-Gen / state-Poss-Acc / understand-Neg

Relevant to Shallow Parsing tasks are especially the problems with inflection. Typically, inflectional affixes have well-defined and transparent grammatical functions such as marking the number or person of the base they are attached to. Structural case markers are especially useful in detecting the grammatical role a constituent plays in a sentence: Nominative ($\emptyset$) marks the subject (i.e. is attached to the word which is the head of the subject NP), whereas Accusative (*-(y)I*) marks the direct object of a sentence. Yet there are two central problems

with their reliability. First of all, verbs select case markers words or morphemes of a language, which are unpredictable related to syntax. [15]. For instance *döv-* to beat and *vur-* to hit verbs gets different case markers. While first, one forces the object that comes before, to get the only accusative, the second one forces to get dative-marked. On the other hand, the dative case marker has several usages. One is to get a role when finding an indirect object, the other is to remarking the inclination object of a sentence. The other issue is in Turkish accusative does not need to be pointed out every time. Accusative is generally used if the object exactly needs to be referenced, or specified [16], [17], [18], [19]. It is likely to use 'the' definite article in English.

Since Turkish is a pro-drop language, it creates further problems for the shallow parsing tasks. Based on inferability some pronouns may be dropped from a sentence. Thus some indicators may be contained in the context hidden, rather than an actual sentence.

Finally, Turkish NLP tasks have another problem such as embedding phrases and sentences. This makes sentences with no ends theoretically possible. There are three types of embedding styles: First one is syntactically marked, which is similar to Indo-European embedding like *that*. Second is unmarked embedded. Last is morphologically marked, which the verb of the embedded sentence is attached with nominalizer [6]. Annotation and parsing processes have major difficulties when facing with non-finite verbs. Furthermore because of Turkish word ordering is free, it is hard to foresee the positions of elements in the sentence.

Especially shallow parsing is a challenging process in Turkish for it is complex and unstable grammar.

3.2.2 Computational Background

In their paper, Eichler and Neumann [20] propose a system for extracting noun groups among various constituents of sentences. They use a sophisticated parser

with several constraints. They do not, however, analyze Turkish data.

In another study, Yildiz et. al. [21] Penn Treebank provides data to external data. They try to automate defining and tagging the chunks over the translation of data to Turkish. For training data, they use conditional random fields (CRF). Aiming to solve the general chunking problem, they report different levels of chunk resolution.

Kutlu & Çiçekli's work is about noun phrase splitting in Turkish [22]. Because of their projects elasticity, their dependency parsers rules are freehanded to comply with complex context. In this way, they arguably get better results for shallow parsing of all phrase types.

Finally, El-Kahlout & Akin [23] argue that chunking is relatively unproblematic for simplex words and for analytic languages like English. However Turkish is morphologically complex language and needs sophisticated features. There are two different approaches proposed for Turkish NLP. in earlier technique chunks are subtracted via the Turkish dependency parsers outcomes. In the after technique, CRF-based chunker which is improved morphological and complex attributes uses annotated Turkish sentences. Results of different experiments show that later used technique has far better performance than early used technique(F-measure: 87.5 in general, and up to 91.95 for verbal chunks).

3.3 Word Sense Disambiguation

3.3.1 Linguistic Background

One of the central foci of literature on Turkish linguistics has been the complexity of Turkish morphology. Turkish is a textbook example for an agglutinative language, i.e. words in their surface form may contain various morphemes, especially suffixes, each of which has a semantic and/or syntactic contribution to the sentential meaning.

In general, a language in its written form gives us orthographic clues about word boundaries but not about the internal makeup of words in their surface form. Moreover, orthography is misleading when multi-word expressions are taken into consideration. In Turkish, the problem of parsing and disambiguation is even more advanced, as not only content words in their base forms but also functional morphemes may be the source of ambiguity which cannot be resolved without the context.

The presence of an excessive amount of suffixes in the Turkish lexicon is closely related to the problem of the storage vs. computation trade-off. In the linguistics literature, there are several approaches to the size of the mental lexicon - the load on memory - and the amount of work to be performed by the computational mechanism - the burden on the processor -. When the parallels among the human brain and computers were drawn at early stages of developments in computer science and linguistics, it was believed by most linguists that human mind has limited storage and an efficient computational mechanism. Only morphemes, i.e. morphologically simplex items such as roots or affixes, were assumed to be stored in lexicon [24]. Developments in computer science made linguists question this belief and most scientists nowadays hold the view that some morphologically complex items are stored rather than assembled in the brain. Therefore, in an agglutinative language like Turkish, considering some morphologically complex word forms as inseparable chunks may help us reduce the rate of ambiguity.

Not only the morphology but also the syntactic structure of Turkish is a fertile source for disambiguity. Turkish is typically treated as a subject-object-verb (SOV) language yet it has a rather free word order. A particular order is chosen mainly on pragmatic grounds [6]. The problem arises due to the fact that, in predicting the correct sense of a given word, WSD needs to take into account the frequency of neighboring words, i.e. to use the word-order as a clue for disambiguation.

Yet another problem of Turkish for WSD studies is languages pro-drop feature,

which is in some cases elements like pronouns can be excluded from the context. This situation can occur when the pronoun is predictable in proper context, and the result of this some elements like a person, or number indicators can be missing in the context whereas they are hidden in actual. Thus in the situations like these missing elements can only be mined through context itself.

In short, many linguistic features in Turkish, such as in which sense a word is used, what the functions of the affixes are, word-order and the antecedent of some agreement relationships are predictable through discourse only. Thus, it is safe to assume that Turkish is a challenging case for WSD tasks.

3.3.2 Computational Background

WSD has some weaknesses about selectable differences between ambiguous words, feature selection, algorithms, and evaluation criteria. In their paper, Orhan and Altan tried to strategize selecting features for WSD in Turkish. The result of the paper shows that while processing sense disambiguation there are a lot of agents that affect Turkish predicates, and also according to the results, selecting the correct and effective features are more important than the algorithms. Therefore, increasing the size of the feature set does not directly affect the performance due to some features being irrelevant [25].

İlgen et al. work on different windowing schemes effect on WSD accuracy in the Turkish language. Turkish lexical sample dataset is used in these experiments. In the experiments, varied machine learning algorithms are used with varied window sizes. According to the results for +-5 window size, Naive Bayes and Functional Tree have better accuracy than other algorithms for verbs and nouns. Finally, it can be argued that smaller window scores are more effective for the disambiguation process. For Turkish WSD data average, 5 window size is the best results are gathered [26].

Altintas et al. investigate the windowing effects on WSD success rates. Authors use Maximum Relatedness Disambiguation algorithm with modification to test the performance of several weighting functions. According to their results, the accuracy of WSD with this approach is increased to 4.24. Therefore, the distance of neighboring words is a significant factor for WSD [27].

In their study İlgen et al. tried to figure out the most appropriate feature groups for Turkish WSD tasks. For this study linguistic dataset which covered multi-sense nouns and verbs. The different feature sets are tested by several machine learning algorithms (Naïve Bayes, IBk, Star, J48, and FT) to find the most effective features. According to the results, preceding and following words of the target word are more effective than other words in the sentence, by increasing the accuracy performance of both verbs and nouns [28].

Chapter 4

Data

4.1 Prework

4.1.1 Data

The dataset is collected from Penn-Treebank corpus and each sentence of this dataset is translated into Turkish [29]. This dataset includes 9,557 sentences, and 73,542 words (including punctuation marks).

The data was stored in a text file as a text format, and it consists of some parts which are morphological analysis, named entity recognition and shallow parsing as shown in Figure 4.1.


```
{turkish=lüks}{morphologicalAnalysis=lüks+ADJ}{metaMorphemes=lüks}{semantics=TUR10-0253770}{namedEntity=NONE}
{propbank=ARG0$TUR10-0668610}{shallowParse=ÖZNE} {turkish=sınıf}{morphologicalAnalysis=sınıf+NOUN+A3SG+PNON+NOM}
{metaMorphemes=sınıf}{semantics=TUR10-0197290}{namedEntity=NONE}{propbank=ARG0$TUR10-0668610}{shallowParse=ÖZNE}
{turkish=otomobil}{morphologicalAnalysis=otomobil+NOUN+A3SG+PNON+NOM}{metaMorphemes=otomobil}{semantics=TUR10-
0520720}{namedEntity=NONE}{propbank=ARG0$TUR10-0668610}{shallowParse=ÖZNE} {turkish=üreticisi}
{morphologicalAnalysis=üretici+NOUN+A3SG+P3SG+NOM}{metaMorphemes=üretici+sH}{semantics=TUR10-1224030}
{namedEntity=NONE}{propbank=ARG0$TUR10-0668610}{shallowParse=ÖZNE} {turkish=geçen}{morphologicalAnalysis=geç+VERB
+POS^DB+ADJ+PRESPART}{metaMorphemes=geç+yAn}{semantics=TUR10-0288360}{namedEntity=NONE}{shallowParse=ZARF_TÜMLECİ}
{turkish=yıl}{morphologicalAnalysis=yıl+NOUN+A3SG+PNON+NOM}{metaMorphemes=yıl}{semantics=TUR10-0676860}
{namedEntity=NONE}{propbank=NONE}{shallowParse=ZARF_TÜMLECİ} {turkish=Birleşik}{morphologicalAnalysis=birleşik
+NOUN+A3SG+PNON+NOM}{metaMorphemes=birleşik}{semantics=TUR10-1229300}{namedEntity=LOCATION}
{shallowParse=DOLAYLI_TÜMLEÇ} {turkish=Devletler'de}{morphologicalAnalysis=devlet+NOUN+PROP+A3PL+PNON+LOC}
{metaMorphemes=devlet+lAr+'DA}{semantics=TUR10-1229300}{namedEntity=LOCATION}{shallowParse=DOLAYLI_TÜMLEÇ}
{turkish=1214}{morphologicalAnalysis=1214+NUM+CARD}{metaMorphemes=1214}{semantics=TUR10-0000010}{namedEntity=NONE}
{propbank=ARG1$TUR10-0668610}{shallowParse=ZARF_TÜMLECİ} {turkish=adet}{morphologicalAnalysis=adet+NOUN+A3SG+PNON
+NOM}{metaMorphemes=adet}{semantics=TUR10-0670790}{namedEntity=NONE}{propbank=ARG1$TUR10-0668610}
{shallowParse=ZARF_TÜMLECİ} {turkish=otomobil}{morphologicalAnalysis=otomobil+NOUN+A3SG+PNON+NOM}
{metaMorphemes=otomobil}{semantics=TUR10-0520720}{namedEntity=NONE}{propbank=ARG1$TUR10-0668610}
{shallowParse=NESNE} {turkish=sattı}{morphologicalAnalysis=sat+VERB+POS+PAST+A3SG}{metaMorphemes=sat+DH}
{semantics=TUR10-0668610}{namedEntity=NONE}{propbank=PREDICATE$TUR10-0668610}{shallowParse=YÜKLEM} {turkish=.}
{morphologicalAnalysis=.+PUNC}{metaMorphemes=.}{semantics=TUR10-1081860}{namedEntity=NONE}{shallowParse=HİÇBİRİ}
```


Figure 4.1: Data content as .train file

4.1.2 Preparing Educational Videos

In this study, most of the sentence labeling was made by non-computer science license students. six detailed videos, which began from introduction and installation of the program to the detailed explanation of labeling morphological analysis, ner, semantic, and shallow parsing, were made for them to understand and carry out the labeling tasks.

4.1.3 Controlling Other Students

At the beginning of this project, three license students at Işık University are assigned. They labeled these 9,557 sentences for NER, Shallow Parsing, and Semantic. To label semantic labeling, first, they had to label morphological analysis as prework. Thus they had to label 38,228 sentences. Controlling this kind of study needs to be done by a computer mostly. An unlabelled word in a sentence algorithm, which investigates all the sentences and returns if any instance is unlabelled did the trick.
Furthermore in every task approximately for every 100 sentences were randomly inspected to check if any slacking had made.

4.2 Tagging

4.2.1 Morphological Disambiguation

Turkish is included in agglutinative language group, in which sentences are structured as unchanging root gets derivational, and inflectional suffixes at the end in general. This ability which allows words to change its part of speech by getting morphemes can convert nouns to verbs or the exact opposite, or adjectives to adverbs. Furthermore, while forming words, some of the letters can be changed, or discarded. For this reason, it is not possible to describe the word and get sense nominees without considering not only its sense but also solving the lemma

of the word from its surface form. Following the translation, corpus has been

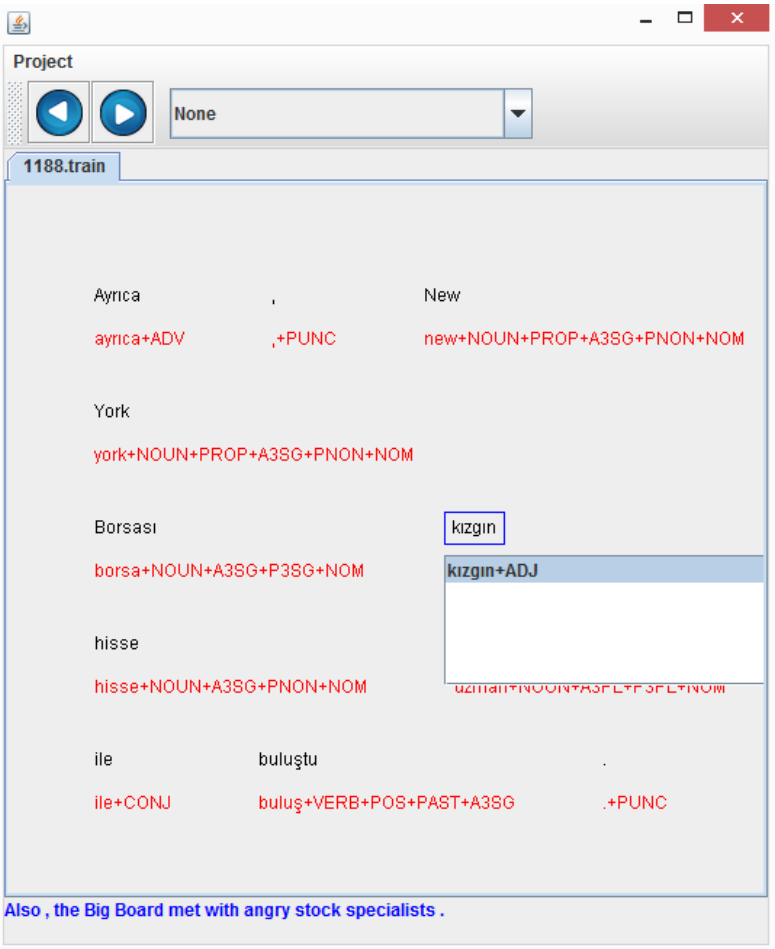

Figure 4.2: Morphological disambiguation tool

morphologically disambiguated. In that work, human annotators selected the accurate morphological parsing from the multiple results of an automatic parser (See Figure 4.2: The Morphological Disambiguation Tool). The representation of morphology and tag group was received from the study [30]. The structure of parser for all words are root, part-of-speech tag, morpheme sets, and '+' signs are the separators for each element.

4.2.2 NER Tagging

In the second stage, 9,557 sentences were annotated manually by using the NER Annotation Tool of Işık University. The tool is shown in Figure 4.3. NER tags

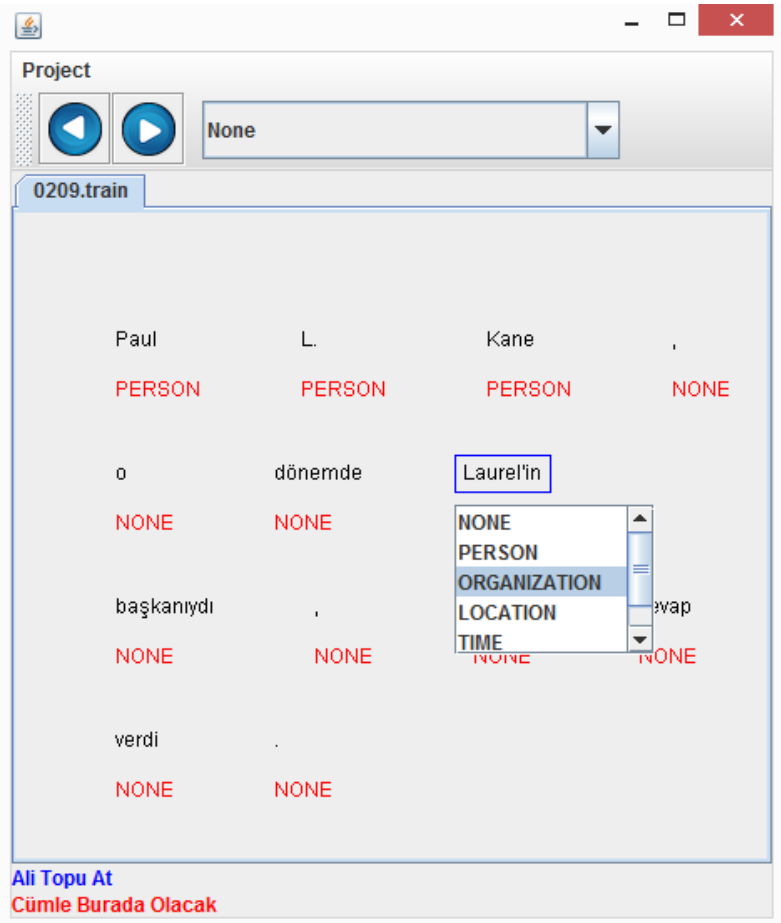

Figure 4.3: Annotation tool for NER

that are used as class labels in this study are: PERSON for person's names, LOCATION for place names, ORGANIZATION for governmental or civilian organizations, firms, associations etc., TIME for specific points in time such as dates, years, hours etc., MONEY for financial amounts or currencies, and NONE for everything else. The data distribution and the class labels for these categories are shown below in Table 4.1.

Table 4.1: Distribution of the data

Label	Count
PERSON	2598
LOCATION	1290
ORGANIZATION	4376
MONEY	2142
TIME	1193
NONE	61894

4.2.3 Shallow Parse Tagging

"Özne (subject)", "Nesne (direct object)", "Yüklem (verb)", "Zarf Tümleci (adverbal clause)", and "Dolaylı Tümleç (indirect object)" which are the elements of a common turkish sentence, are in this dataset, whose implementation is shown in Table 4.2. In this stage, 9,557 sentences were annotated manually by using

Table 4.2: Tags and their occurrences

Tag	Occurrence (word)
HIÇBIRI	6509
NESNE	15685
ÖZNE	14801
YÜKLEM	11701
ZARF TÜMLECI	13321
DOLAYLI TÜMLEÇ	11701

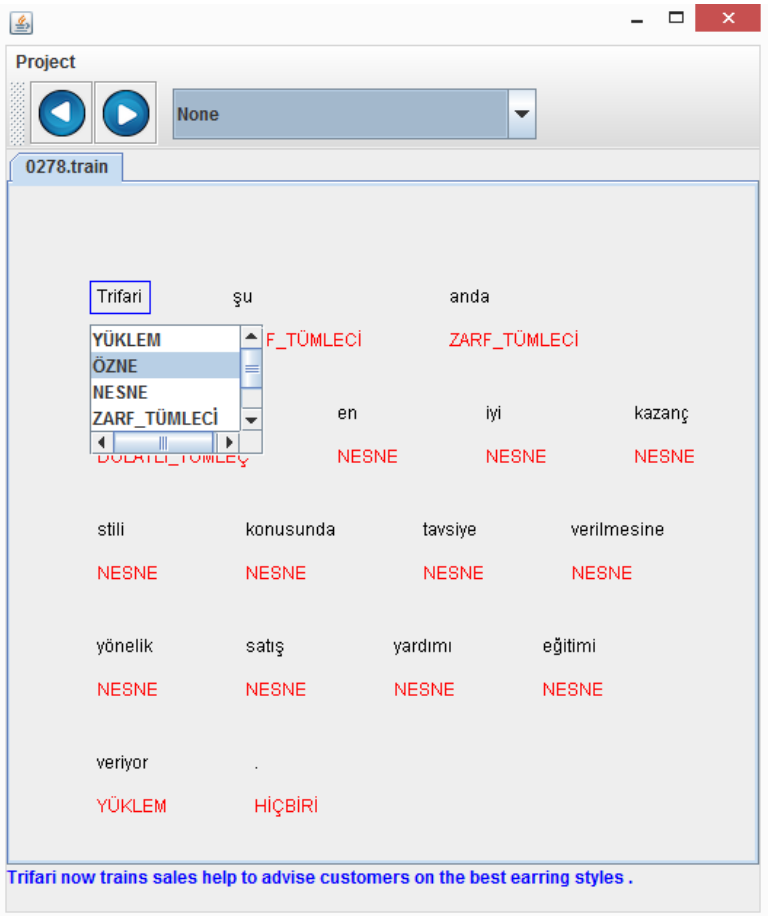

Figure 4.4: Annotation tool for Shallow Parse

the Shallow Parsing Annotation Tool of Işık University. Figure 4.4 illustrates the annotation tool that is used.

4.2.4 Word Sense Disambiguation Tagging

The most frequently used class labels, their counts and meanings are given in (Table 4.3). There are no transposed sentences in the dataset. In the annotation, NLP Tool of Işık University was used. In the annotation screen, there are some

Table 4.3: The most used class labels

Label	Count	Sense
TUR10-1081860	1282	nokta
TUR10-0000000	987	özel isim
TUR10-0820240	418	virgül
TUR10-0775480	268	tırnak işareti
TUR10-0000010	235	bir tam sayı
TUR10-0105580	171	herhangi bir varlığı belirsiz olarak gösteren sayı
TUR10-0816400	159	bir bağ olduğunu anlatan bir söz
TUR10-0000020	157	reel sayı
TUR10-0178920	146	da / de
TUR10-0120760	92	bu

possible meanings under each word of each sentence. For tagging the words, the person (tagger) needs to choose the appropriate meaning for each word considering the items from a combo box as illustrated in (Figure 4.5). When the tagger presses the "next" button, his/her annotation is saved and the tagger can pass to the next one to annotate it. Annotated dataset and source codes are freely accessible[1].

[1] http://haydut.isikun.edu.tr/nlptoolkit.html

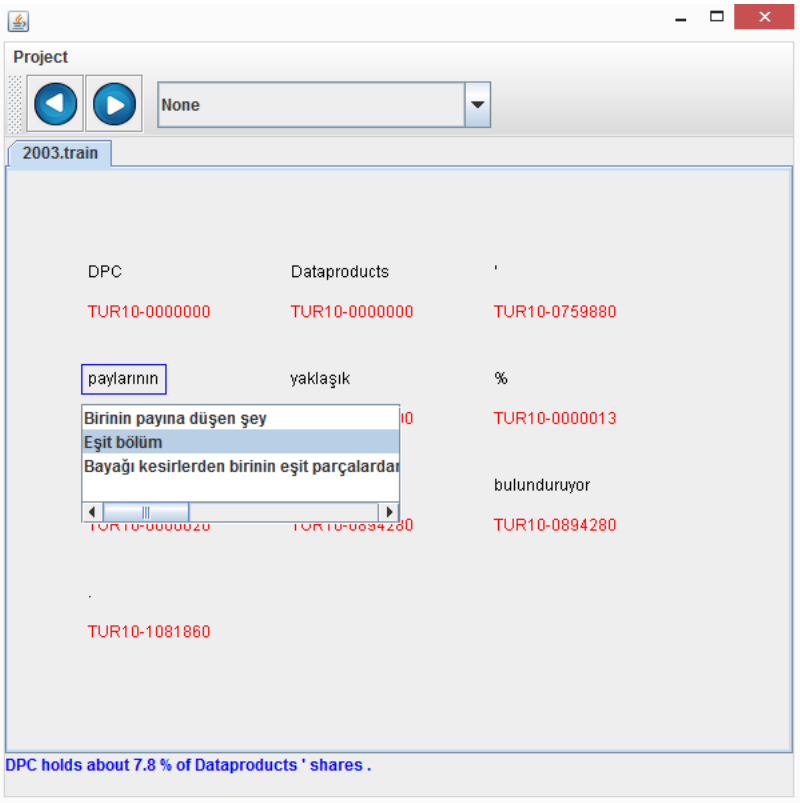

Figure 4.5: Annotation screen for WSD

Chapter 5

Algorithms

The six known classification algorithms we use, are:

Dummy classifiers decisions are prior class probability based without input unaffected. It provides a baseline for other classifiers. All test samples are assigned to the class with maximum precedence.

Naive Bayes classifier, which depends on Bayes' Rule, that says "If there are two, random, jointly distributed variables with one variable is known, others probability can be found by computing given value" where all of the features are assumed to be Gaussian distributed [31] and each of them are independent of other features.

C 4.5 is an entropy using decision tree. Entropy is an indicator that shows how pure or impure the dataset is. This type of decision tree is a recursive algorithm that keeps splitting data until sets are pure. This cause overfitting. To prevent overfitting C4,5 uses pruning which stops run if accuracy decreases.

K-Nearest Neighbor is a classification algorithm which says that in a 'n' element cluster KNN classifies the input by the majority of k^{th} nearest neighbors votes, while all neighbors have equal vote [31], that uses the Euclidean distance.

Random Forest is a combined decision tree which develops bagging idea with feature randomization for each decision node [32] and calls them as weak learners. In

the prediction cycle, with the help of committee-based procedures, weak learners are united.

Rocchio is emerged from vector space model and based on Naive Bayes algorithm with relevance feedback which tries to figure out optimal query vector that maximizes resemblance to related documents while minimizes resemblance to irrelate documents.

Chapter 6

Features

Two types of attributes are used in this thesis. First type is binary which returns either true, or false, and second one is discrete which returns either string or "null". 21 features are used for shallow parsing, and word sense disambiguation. 25 features are used for Named entity recognition. All features with alphabetical order are as follows:

CaseAttribute (`CA`) is a discrete feature for a given word. If the last inflectional group of the word contains case information, the attribute will have that case value. Otherwise the attribute will have the value **null**.

IsAdjective Attribute (`IAA`) is a binary feature, which shows a word is used whether as adjective or not.

IsAuxillaryVerb Attribute (`IAVA`) is a binary feature, that checks the word is whether an auxiliary verb or not. This is one of the strong features because it may be useful in identifying the word as a (part of) predicate ("yüklem").

IsCapitalAttribute (`ICA`) is a binary feature, which checks each word in a sentence and If the first character is uppercase, then returns **true**, otherwise **false**.

IsDateAttribute (`IDA`) is a binary feature, that checks the word is written whether in date format or not. If the word is in date format, feature returns **true**, otherwise **false**.

IsFractionAttribute (`IFA`) is a binary feature, that checks the word is whether a fractional number. If the word is a fractional number, feature returns **true**, otherwise **false**.

IsHonorificAttribute (`IHA`) is a binary feature, that checks each word in a sentence. If the word equals to one of the titles “bay”, “bayan”, “sayın”, “dr.”, “prof.” or “doç.”, feature returns **true**, otherwise **false**.

IsLocationGazetteerAttribute (`ILGA`) is a binary feature, that checks each word in the sentence. If word is in “gazetteer-location.txt“, feature returns **true**, otherwise **false**.

IsMoneyAttribute (`IMA`) is a binary feature, that checks the word in a sentence is whether denotes a currency. If the word equals to one of the money units “doları”, “lirası” or “Avro”, the feature returns **true**, otherwise **false**.

IsNumberAttribute (`INA`) is a binary feature, that checks whether the word has num (number) tag.

IsOrganizationAttribute (`IOA`) is a binary feature, that checks each word in the sentence. If the word equals to one of the titles “corp”, “inc.”, or “co.”, the feature returns **true**, otherwise **false**.

IsOrganizationGazetteerAttribute (`IOGA`) is a binary feature, that checks each word in the sentence. If word is in “gazetteer-organization.txt“, the feature returns **true**, otherwise **false**.

IsPersonGazetteerAttribute (`IPGA`) is a binary feature, that checks each word in the sentence. If word is in “gazetteer-person.txt“, feature returns **true**, otherwise **false**.

IsProperNounAttribute (`IPNA`) is a binary feature, that checks the word has whether prop (proper name) tag.

IsRealAttribute (`IRA`) is a binary feature, that checks each word in a sentence is whether a real number or not. If the word is a real number, the feature returns **true**, otherwise **false**.

IsTimeAttribute (`ITA`) is a binary feature, that checks the word is written whether in time format. If the word is in time format, the feature returns **true**, otherwise **false**.

LastIGContainsPossessiveAttribute (`LICPA`) is a binary feature. If the last inflectional group of the word contains possessive information, the feature will return **true**, otherwise return **false**.

LastIGContainsTagAblative Attribute (`LICTAblativeA`) is a binary feature, that checks the word is whether ablative case or not.

LastIGContainsTagAccusative Attribute (`LICTAccusativeA`) is a binary feature, that checks the word is whether accusative case or not.

LastIGContainsTagGenitive Attribute (`LICTGenitiveA`) is a binary feature, that checks the word is whether genitive case or not.

LastIGContainsTagInstrumental Attribute (`LICTInstrumentalA`) is a binary feature, that checks the word is whether instrumental case or not.

MainPosAttribute (`MPA`) is a discrete feature, which returns the main part of speech of the word.

PredicateAttribute (`PA`) is a discrete feature, which searches sentence for each word to find a word or words that has “verb“ in rootpos and lastpos. If it finds such a word in sentence returns it as predicate. If it finds more than one word, it checks the distances for current highlighted word.

RootFormAttribute (`RFA`) is a discrete feature, which returns the root form of a given word.

RootPosAttribute (`RPA`) is a discrete feature, which returns the part of speech of the root form of a given word.

Chapter 7

Experiments

At first, the program was run with all features for all classifiers. Then feature selection was applied and re-run with selected features. All the received data are as follows for 7.3, 7.4 and 7.5 accordingly.

The dataset of NLP toolkit of Isik University is used. The annotation was done manually by several annotators. Some annotators were experts on the grammar of Turkish while others are native speakers of Turkish with no special experience in this area. Therefore, it is possible that they might annotate some words wrong in complicated sentences.

7.1 Experiment Setup

The setup of the experiments are as follows:

Window size that is used is two for all experiments.

In SP and NER experiment runs datas are set as stratified 10 fold cross-validation, on the other hand on WSD experiment runs data is set as 10 fold cross-validation.

All NER, SP, and WSD datasets are changed to indexed from discrete and normalized.

For feature selection, only “Forward Selection“ is applied, due to lack of time.

Rocchio classifiers Distance Attribute is set to “Euclidean Distance“ for NER, SP, and WSD.

C4.5 classifiers Prune Attribute is set to “true“ for NER and SP, whereas “false“ for WSD.

KNN classifiers Distance Attribute is set to ‘Euclidean Distance“, and K attribute is selected from the set to {1, 3, 5, 7, 9, 11, 13} for NER, and WSD, whereas is set to {1, 3, 5, 7, 9, 11, 13, 15, 17, 19} .

Random Forest classifiers Ensemble Attribute is set to {5, 10, 15, 20, 25, 30, 35}, and Subsize Attribute is set to {1, 2, 4, 8, 16, 32, 64} for NER, SP, and WSD.

7.2 Inter-annotator Agreement

The inter-annotator-agreement measure which is for the assessment of the annotated dataset is used in this thesis. Two separate groups annotated the same sentences. However, only 100 of the 9,557 sentences were re-annotated because of insufficient teams. Table 7.1 shows the expected and actual agreement scores. This agreement is based on the assumption of annotators do the annotation randomly.

Table 7.1: Inter-Annotator Agreement

Layer	**Agreement**	**Expected Agreement**
NER	97.5	16.7
SP	79.0	16.7
WSD	78.5	54.5

7.3 NER

The proposed approach is evaluated by using stratified 10-fold cross-validation on the data. The accuracies are obtained from each classifier are shown in Tables.

For window size two, classifiers look up to both two predecessors of the word, the word itself and two successors of the word. Dummy classifiers feature selection

Table 7.2: NER: Dummy classifier with all features

Results
87.08 ± 0.01

result for NER is 87.08 ± 0.01 with no attribute selections or word selections. Because the result of Dummy is irrelevant with features.

Tuning of Naive Bayes result for this dataset is worse than Dummy classifier. On

Table 7.3: NER: Naive Bayes classifier with all features

Results
64.81 ± 0.38

another dataset, it might get better results. The most selected word is the word

Table 7.4: NER: Naive Bayes Feature Selection

Attribute	Words	Results
1 IAA	3	95.83 ± 0.17
2 IAVA	5	
3 IFA	2, 4	
4 IHA	2, 3, 4	
5 ILGA	3, 5	
6 IMA	2, 3, 4, 5	
7 IOA	4	
8 IPNA	2, 3, 4	
9 IRA	1, 3	
10 ITA	3, 4	
11 LICPA	3	
12 RFA	3	

itself, and the most selected attribute is “IsMoneyAttribute“ for Naive Bayes.

Tuning of Rocchio result for this dataset is worse than Dummy classifier. On another dataset, it might get better results.

The most selected word is the word itself, and the most selected attribute is “RootFormAttribute“ for Rocchio.

Table 7.5: NER: Rocchio classifier with all features

Distance	Results
Euclidian	72.38 ± 0.70

Table 7.6: NER: Rocchio Feature Selection

Attribute	Words	Results
1 IPNA	3	91.64 ± 0.21
2 ITA	3	
3 MPA	1	
4 RFA	2, 3, 5	
5 CA	2	

Tuning of C4.5 result for this dataset is better than Dummy classifier.

Table 7.7: NER: C 4.5 classifier with all features

Pruning	Results
true	94.37 ± 0.26

KNN has the best tuning results all classifiers for this dataset.

Table 7.8: NER: KNN classifier with all features

k	Distance	Results
1	Euclidian	94.40 ± 0.21
3		94.87 ± 0.08
5		94.90 ± 0.15
7		94.80 ± 0.21
9		94.69 ± 0.17
11		94.62 ± 0.17
13		94.51 ± 0.18

Tuning of Random Forest result for this dataset is better than Dummy classifier and has the second-best results. Best 30 results are shown in the table.

Table 7.9: NER: Random Forest classifier with all features

Ensemble	Subsize	Results		Ensemble	Subsize	Results
5	8	94.18 ± 0.29		20	32	94.60 ± 0.25
5	16	94.16 ± 0.27		20	64	94.57 ± 0.24
5	32	94.38 ± 0.26		25	4	87.42 ± 0.12
5	64	94.36 ± 0.25		25	8	94.40 ± 0.26
10	8	94.29 ± 0.27		25	16	94.36 ± 0.23
10	16	94.26 ± 0.23		25	32	94.62 ± 0.27
10	32	94.49 ± 0.24		25	64	94.59 ± 0.27
10	64	94.48 ± 0.23		30	8	94.41 ± 0.27
15	8	94.34 ± 0.25		30	16	94.38 ± 0.26
15	16	94.32 ± 0.24		30	32	94.62 ± 0.26
15	32	94.57 ± 0.25		30	64	94.61 ± 0.26
15	64	94.55 ± 0.25		35	8	94.41 ± 0.25
20	4	87.42 ± 0.12		35	16	94.39 ± 0.23
20	8	94.37 ± 0.27		35	32	94.63 ± 0.25
20	16	94.34 ± 0.24		35	64	94.62 ± 0.24

7.4 Shallow Parse

In the experiment phase, we adopted our dataset in Turkish language. Therefore, we had the advantage of a stratified 10-fold cross validation for our data.

For window size two, classifiers look up to both two predecessors of the word, the word itself and two successors of the word. Total of 21 attributes are used in Shallow Parse, "*gazetteer*" and "predicate" attributes are not included.

Dummy classifiers feature selection result for shallow parse is 22.04 ± 0.01 with

Table 7.10: SP: Dummy classifier with all features

Results
22.04 ± 0.01

no attribute selections or word selections. Because the result of Dummy is irrelevant with features.

Naive Bayes has the best tuning results all classifiers for this dataset.

There are two most selected words for feature selection: the word itself, and

Table 7.11: SP: Naive Bayes classifier with all features

Results
64.95 ± 0.30

Table 7.12: SP: Naive Bayes Feature Selection

Attribute	Words	Results
1 IAA	2, 4	68.05 ± 0.33
2 IAVA	1	
3 IFA	5	
4 IHA	2, 3, 4	
5 IOA	1, 3	
6 LICTGenitiveA	1, 5	
7 RFA	3, 4, 5	
8 CA	2, 4, 5	

first predecessor, and most selected attributes are "IsHonorificAttribute", "RootFormAtribute", and "CaseAttribute" for Naive Bayes.

Tuning of Rocchio result for this dataset is better than Dummy classifier.

Table 7.13: SP: Rocchio classifier with all features

Distance	Results
Euclidian	45.19 ± 0.30

The most selected words are the word itself. There are two most selected at-

Table 7.14: SP: Rocchio Feature Selection

Attribute	Words	Results
1 MPA	1, 3	47.96 ± 0.25
2 PA	3	
3 RFA	3, 4	
4 CA	3	

tributes in feature selection: "MainPosAttribute", and "RootFormAttribute" for Rocchio.

Tuning of C4.5 result for this dataset is better than Dummy classifier.

Tuning of KNN results are better than Dummy, and the second best of all clas-

Table 7.15: SP: C 4.5 classifier with all features

Pruning	Results
true	51.07 ± 0.43

Table 7.16: SP: KNN classifier with all features

k	Distance	Results
1	Euclidian	57.47 ± 0.41
3		58.99 ± 0.20
5		60.73 ± 0.32
7		61.45 ± 0.35
9		61.69 ± 0.35
11		61.93 ± 0.48
13		62.02 ± 0.54
15		63.27 ± 0.53
17		63.31 ± 0.59
19		63.29 ± 0.39

sifiers for this dataset.

Tuning of Random Forest result for this dataset is better than Dummy classifier.

Table 7.17: SP: Random Forest classifier with all features

Ensemble	Subsize	Results		Ensemble	Subsize	Results
5	8	33.27 ± 0.57		20	64	45.52 ± 0.69
5	16	35.93 ± 0.60		25	4	24.60 ± 0.19
5	32	48.33 ± 0.44		25	8	33.30 ± 0.53
5	64	44.47 ± 0.57		25	16	36.00 ± 0.59
10	8	33.29 ± 0.57		25	32	49.78 ± 0.40
10	16	36.01 ± 0.65		25	64	45.63 ± 0.62
10	32	49.10 ± 0.47		30	4	24.60 ± 0.19
10	64	45.09 ± 0.64		30	8	33.32 ± 0.53
15	8	33.27 ± 0.56		30	16	36.01 ± 0.59
15	16	35.97 ± 0.60		30	32	49.81 ± 0.44
15	32	49.48 ± 0.52		30	64	45.62 ± 0.55
15	64	45.29 ± 0.64		35	8	33.31 ± 0.52
20	8	33.31 ± 0.53		35	16	36.00 ± 0.59
20	16	36.04 ± 0.57		35	32	49.84 ± 0.51
20	32	49.64 ± 0.47		35	64	45.68 ± 0.56

Best 30 results are shown in the table.

7.5 Word Sense Disambiguation

In this thesis, the proposed approach is evaluated by using 10-fold cross-validation on the data. The accuracies are obtained from each classifier are shown in Tables. For window size two, classifiers look up to both two predecessors of the word, the word itself and two successors of the word.

Total of 22 attributes are used in Word Sense Disambiguation, "*gazetteer*" attributes are not included.

Table 7.18: WSD: Dummy classifier with all features

Results
68.17 ± 0.33

Tuning accuracy of Naive Bayes is worse than Dummy classifier for this dataset. However, it might change on another dataset.

Tuning of Rocchio result for this dataset is worse than Dummy classifier. On another dataset, it might get better results.

Table 7.19: WSD: Naive Bayes classifier with all features

Results
61.04 ± 0.55

Tuning of C4.5 result for this dataset is better than Dummy classifier.

Table 7.20: WSD: Rocchio classifier with all features

Distance	**Results**
Euclidian	63.27 ± 0.47

Tuning of KNN result is better than Dummy and has the best of all classifiers for this dataset.

Table 7.21: WSD: C 4.5 classifier with all features

Pruning	**Results**
false	70.97 ± 0.34

Table 7.22: WSD: KNN classifier with all features

k	**Distance**	**Reults**
1	Euclidian	69.26 ± 0.51
3		69.94 ± 0.34
5		70.47 ± 0.45
7		70.56 ± 0.52
9		70.39 ± 0.75
11		70.38 ± 0.73
13		70.26 ± 0.62

Tuning of Random Forest result for this dataset is worse than Dummy classifier. On another dataset, it might get better results. Best 30 results are shown in the table.

Table 7.23: WSD: Random Forest classifier with all features

Ensemble	**Subsize**	**Results**		**Ensemble**	**Subsize**	**Results**
5	1	67.74 ± 0.38		20	1	67.84 ± 0.41
5	2	67.70 ± 0.34		20	2	67.81 ± 0.61
5	4	67.66 ± 0.37		20	4	67.84 ± 0.68
5	8	66.96 ± 0.67		20	8	67.12 ± 0.57
5	16	66.89 ± 0.48		20	16	67.14 ± 0.42
10	1	67.89 ± 0.52		25	1	67.95 ± 0.48
10	2	67.82 ± 0.48		25	2	67.78 ± 0.54
10	4	67.76 ± 0.59		25	4	67.78 ± 0.47
10	8	67.08 ± 0.36		25	8	67.22 ± 0.15
10	16	66.97 ± 0.55		25	16	67.10 ± 0.40
15	1	67.89 ± 0.37		30	1	67.91 ± 0.58
15	2	67.74 ± 0.30		30	2	67.81 ± 0.41
15	4	67.81 ± 0.63		35	4	67.88 ± 0.63
15	8	67.12 ± 0.50		35	8	67.27 ± 0.62
15	16	67.10 ± 0.46		35	16	67.19 ± 0.54

Chapter 8

Conclusion

In this thesis, we applied Named Entity Recognition, Shallow Parse and Word Sense Disambiguation tests with different discrete classifiers and parameters. For this, we firstly mark the 9,557 sentences with seven different annotators using the NLP toolkit. Then we test six different (Dummy, NB, Rocchio, C4.5, KNN, RF) with all features included and applied feature selection.

The first of these problems, Named Entity Recognition, defines the word or group of words in terms of either the person, the organization, the location, a time, or money. The obtained rankings of the algorithms for our parameters are "Naive Bayes Feature Selection (95.83), KNN (94.9), Random Forest (94.63), C4.5 (94.37), Rocchio Feature Selection (91.64), Dummy (87.08), Rocchio (72.38), Naive Bayes (64.81)". For Named Entity Recognition when feature selection is applied, results of Naive Bayes and Rocchio are increased drastically.

The second problem, Shallow Parsing, recognizes the word or phrase by the subject, the direct object, the indirect object, the adverbial clause, or the predicate. The obtained rankings of the algorithms for our parameters are "Naive Bayes Feature Selection (68.05), Naive Bayes (64.95), KNN (63.31), C4.5 (51.07), Ranedom Forest (49.84), Rocchio Feature Selection (47.96), Rocchio (45.19), Dummy (22.04)". For Shallow parsing when feature selection is applied, results of Naive Bayes and Rocchio are increased very little.

The third problem, Word Sense Disambiguation, tries to find the correct meaning of the word or phrase. The obtained rankings of the algorithms for our parameters are C4.5 (70.97), KNN (70.56), Dummy (68.17), RF (67.95), Rocchio (63.27), NB(61.04).

KNN gave good results for all three problems. Gazetteer and predicate features which are generated only for NER are selected very little in feature selections. In general, the most chosen feature is “Root Form Attribute“.

In the studies to be carried out in the future period, besides the discrete features, better results can be obtained by adding continuous features.

References

[1] B. Ertopçu, A. Kanburoğlu, O. Topsakal, O. Açıkgöz, A. Gürkan, B. Özenç, I. Çam, B. Avar, G. Ercan, and O. T. Yıldız, "A new approach for named entity recognition," 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8093439

[2] O. Topsakal, O. Açıkgöz, A. Gürkan, A. Kanburoğlu, B. Ertopçu, B. Özenç, I. Çam, B. Avar, G. Ercan, and O. T. Yıldız, "Shallow parsing in Turkish," 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8093440

[3] O. Açıkgöz, A. Gürkan, B. Ertopçu, O. Topsakal, B. Özenç, A. Kanburoğlu, I. Çam, B. Avar, G. Ercan, and O. T. Yıldız, "All-words word sense disambiguation for Turkish," 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8093442

[4] O. Yıldız, K. Ak, G. Ercan, O. Topsakal, and C. Asmazoğlu, "A multilayer annotated corpus for turkish," 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8374369

[5] E. Okur, *Named Entity Recognition for Turkish Microblog Texts Using Semi-Supervised Learning with Word Embeddings*, 2011.

[6] E. T. Erguvanlı, *The function of word order in Turkish Grammar*. Oxford: Oxford University Press, 1984.

[7] E. Sezer, "On non-final stress in Turkish," *Journal of Turkish Studies*, vol. 5, pp. 61–69, 1981.

[8] S. Cucerzan and D. Yarowsky, "Language independent named entity recognition combining morphological and contextual evidence," in *Proceedings of the Joint SIGDAT Conference on EMNLP and VLC*, 1999.

[9] G. Tür, D. Hakkani-Tür, and K. Oflazer, "A statistical information extraction system for Turkish," *Natural Language Engineering*, vol. 9, pp. 181–210, 2003.

[10] G. Çelikkaya, D. Torunoğlu, and G. Eryiğit, "Named entity recognition on real data: a preliminary investigation for Turkish," in *7th International Conference on Application of Information and Communication Technologies*, 2013.

[11] G. A. Şeker and G. Eryiğit, "Initial explorations on using crfs for Turkish named entity recognition," in *COLING*, 2012.

[12] S. Tatar and İlyas Çiçekli, "Automatic rule learning exploiting morphological features for named entity recognition in Turkish," *Journal of Information Science*, vol. 37, pp. 137–151, 2011.

[13] D. Küçük and R. Steinberger, "Experiments to improve named entity recognition on Turkish tweets," *arXiv:1410.8668*, 2014.

[14] A. Carnie, *Syntax: A Generative Introduction.* Blackwell, 2006.

[15] N. Chomsky, *Lectures on Government and Binding.* Dordrecht: Foris, 1981.

[16] G. Lewis, *Turkish Grammar.* Oxford: Clarendon, 1967.

[17] J. Kornfilt, "Case marking, agreement, and empty categories in Turkish," Ph.D. dissertation, Harvard University, 1984.

[18] J. Kornfilt, *Turkish.* London: Routledge, 1997.

[19] M. Dede, "Definiteness and referentiality in Turkish verbal sentences," in *Studies in Turkish linguistics.* Amsterdam: Benjamins, 1986, pp. 147–164.

[20] K. Eichler and G. Neumann, "Bootstrapping noun groups using closed-class elements only," in *LWA2010 - Workshop-Woche: Lernen, Wissen Adaptivitaet*, 2010.

[21] O. T. Yıldız, E. Solak, R. Ehsani, and O. Görgün, "Chunking in Turkish with conditional random fields," in *CICLing*, 2015, pp. 173–184.

[22] M. Kutlu and I. Cicekli, "Noun phrase chunking for Turkish using a dependency parser," in *ISCIS*, 2015, pp. 381–391.

[23] I. D. El-Kahlout and A. A. Akın, "Turkish constituent chunking with morphological and contextual features," in *CICLing*, 2013, pp. 270–281.

[24] D. Mackay, "Lexical insertion, inflection, and derivation: creative processes in word production," *Journal of Psycholinguistic Research*, vol. 8, pp. 477–498, 1979.

[25] Z. Orhan and Z. Altan, "Impact of feature selection for corpus-based wsd in Turkish," in *MICAI 2006: Advances in Artificial Intelligence*, vol. 4293, 2006.

[26] B. İlgen, E. Adalı, and A. Tantuğ, "A comparative study to determine the effective window size of Turkish word sense disambiguation systems," in *Information Sciences and Systems*, vol. 264, 2013.

[27] E. Altıntaş, E. Karslıgil, and V. Coşkun, "The effect of windowing in word sense disambiguation," in *Computer and Information Sciences*, vol. 3733, 2005.

[28] B. İlgen, E. Adalı, and A. C. Tantuğ, "The impact of collocational features in Turkish word sense disambiguation," in *16th International Conference on Intelligent Engineering Systems*, 2012.

[29] O. T. Yıldız, E. Solak, O. Görgün, and R. Ehsani, "Constructing a Turkish-English parallel treebank," in *Proceedings of the 52nd Annual Meeting of the*

Association for Computational Linguistics. Baltimore, Maryland: Association for Computational Linguistics, June 2014, pp. 112–117.

[30] K. Oflazer, B. Say, D. Hakkani-Tür, and G. Tür, "Building a Turkish treebank," in *Building and exploiting syntactically-annotated corpora*. Dordrecht: Kluwer, 2003.

[31] E. Alpaydın, *Introduction to Machine Learning*, 3rd ed. The MIT Press, 2010.

[32] L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5–32, 2001.