BARIŞ CAN ERKOÇ

RULE-BASED CHUNKING OF SEMANTIC ROLES FOR TURKISH

BARIŞ CAN ERKOÇ

IŞIK UNIVERSITY
2018

BARIŞ CAN ERKOÇ

M.S. Thesis

2018

RULE-BASED CHUNKING OF SEMANTIC ROLES FOR TURKISH

BARIŞ CAN ERKOÇ
B.S., Computer Engineering, IŞIK UNIVERSITY, 2015

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
2018

IŞIK UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

RULE-BASED CHUNKING OF SEMANTIC ROLES FOR TURKISH

BARIŞ CAN ERKOÇ

APPROVED BY:

Prof. ERCAN SOLAK (Thesis Supervisor)	Işık University	_____
Assoc. Prof. OLCAY TANER YILDIZ	Işık University	_____
Assoc. Prof. ARZUCAN ÖZGÜR	Boğaziçi University	_____

APPROVAL DATE: ..../..../....

RULE-BASED CHUNKING OF SEMANTIC ROLES FOR TURKISH

Abstract

In our work, we approached to semantic role labeling from a different angle. Contrary to related works, which focused on determining single role like noun phrase or predicate, we worked on all of the roles. We claim that, morphological analysis of a word and its context can be useful for semantic role labeling task. For that, we first determine the possible semantic chunk boundaries by examining the morphological analysis of words and their contexts. For further improvement in determining the boundaries, we do the first process with the combination of the morphological analysis and the boundary output from the first pass. We use these boundaries to create semantic chunks and labeled them according to their content.

Keywords: Turkish dependency parsing, semantic role labeling, rule based dependency parsing

Table of Contents

List of Tables

List of Figures

List of Abbreviations

NLP	**N**atural **L**anguage **P**rocessing
SRL	**S**emantic **R**ole **L**abeling
VP	**V**erb **P**hrase
NP	**N**oun **P**hrase

Chapter 1

Introduction

The parsing is a process to analyze a sentence's elements according to the language's grammar. Language in question can be a natural language or a computer language. For natural language, there are two different parsing types, constituency parsing and dependency parsing. While constituency parsing is focused on the relations of the constituents of the sentences like noun phrases and predicates, dependency parsing is focused on the relations of the words.

By parsing, one can extract named entities, like locations, persons, coorporation names etc., translate from one language to another, if there is a high accuracy parser for each language, or create automated systems that can analyze an input and give proper output, like automated Q&A or write anything on a given context.

Task of dependency parsing is to figure out the relations between pairs of words of a sentence. These relations are called dependency relations and consists of word pairs and label. Each pair has elements named head and dependent. The head is assumed to be the representative of its dependents. Each word has exactly one head, if it is not the head of the sentence, if so it has none. Each head can have multiple dependents.

The programs whose task's are parsing are called parsers. Parsers enables us to figure out the semantic and syntactic structures of a language. The most used method in parsers is statistical approach. So first, parser trained on some data

set that has been parsed by hand to figure out the frequencies of the elements in it. Then the parser runs on test data, which is also parsed by hand, to determine the accuracy of the system. This method is very useful for learning the structure of a language but there is a problem that must be dealt with, which is overfitting to data. Also statistical methods depend on larger sets of labeled data to figure out the structure.

Also there are some rule-based approaches [1] [29]. This method, unlike the statistical approaches, require deep understanding of grammar of the language in question to make decisions, rules. Rule-based approaches can work on a small set of labeled data and use this data only for checking the validity and accuracy of the rules created, not for the decision. Apart form the statistical approaches, rule-based approaches do not deal with overfitting problem but accuracy heavily depends on knowledge of the language in question.

Current approaches for dependency parsing of Turkish are all statistical [21] [4]. As mentioned before, this approach requires large data set for training and currently, the Turkish dependency data is only about 4000 sentences, which is far too small when compared to other languages and it contains many sentences that are somewhat synthetic which does not represent the natural use of the language.

In natural language processing, NLP, there is a task related to the semantics. Simply a sentence structure is made of combination of the small semantic elements known as words. There are other elements in the sentence structure at higher semantic levels named semantic roles. The related task to find those elements is named semantic role labeling, SRL. This task's purpose is finding semantic roles in the sentence like subject, object and predicate.

We propose 3 step rule-based approach for dependency parsing of Turkish. In first step, we extract semantic chunk boundaries from the morphological features and POS tags. In second step, we label these semantic chunks. In third step, we first create dependency relations between the semantic roles then we will create dependecy relations between each semantic role's own words.

This thesis is organized as follows. In Chapter 2 we give information about Turkish morphology. In Chapter 3 we examine the existing literature on chunking, SRL and dependency parsing. We detail our approach in Chapter 4. We cover our data preparations in Chapter 5 and the rules in Chapter 6. In Chapter 7 we present our experimental results and feature selection process and discuss our results in Chapter 8 then conclude our work in Chapter 9.

Chapter 2

Preliminaries

2.1 Turkish Morphology

Morphology is the study of identifying the structure of the words by analyzing how new words are added into a language or analyzing how a word's form changes when used in a sentence in different ways. Native speakers of any language heuristically know how to create and add a new words to their native language, can judge a word, by examining the form, and decide whether the word can be a part of the speaker's native language or not[19].

Problem of defining what makes word a word is the problem of morphology. Words are defined as a combined structure of smallest meaningful elements of a language. These elements are called morphemes. In English, because their roots are the smallest meaningful element of the word, words like "home", "come" or "understand" are morphemes. Also prefixes such as re-, pre-, or un- and suffixes such as -er, -ful, or -ment are morphemes. By the definition of morphemes given above, not only the word "state" is a morpheme but so is the suffix -ment in word "statement" alongside with the morpheme "state". Some examples of the English and Turkish morphemes are presented in the Table 2.1 .

In Turkish, construction of words is done by suffixation [9] . Suffixation is divided into two categories, derivational and inflectional. Suffixes can be added directly to the root or can be added to complex root-suffix structures. Every morphological

English Morphemes		Turkish Morphemes	
re-	**re**capture	**-lar**	araba**lar**
pre-	**pre**process	**-a**	ev**e**
un-	**un**certain	**-de**	ev**de**
-er	teach**er**	**-sız**	kullanış**sız**
-ful	thank**ful**	**-li**	ev**li**
-ment	state**ment**	**-sel**	belge**sel**

Table 2.1: Some of the morphemes of English (first 2 columns) and Turkish (last 2 columns)

element whose structure can be changed by suffixation process, like mentioned above, is called stem. Few steps of word construction based on stem "korkmak" is presented in the Table 2.2 .

korkmak	to fear
korku	fear
korkusuz	fearless
korkusuzlaşmak	to become fearless
korkusuzlaşmış	one who has become fearless

Table 2.2: Evaluation of the kork-, (to) fear, stem by adding new suffixes to newly produced stem

Derivational suffixes change the lexical meaning of the word they are attached to, creating new lexical item. In Table 2.2 verb "korkmak" (to fear) becomes "korku" (fear) when suffix "-I" is attached to it ('I' can be i, ı, u, ü). In derivation, POS can be preserved (like in verb "şaşmak" to verb "şaşakalmak") or not (like in verb "korkmak" to nominal "korku").

Inflectional suffixes create the relation between the words of the sentence in forms of case, person, tense, possessor etc. They are categorized as nominal inflectional suffixes and verbal inflectional suffixes.

Nominal inflection suffixes are plural, possessive and case suffixes (Table 2.3) while verbal inflection suffixes are, for finite verbs, voice suffixes (causative, passive, reflexive, reciprocal suffixes), negative, copular, person (Table 2.4), tense, aspect, modality markers (Table 2.5) and for non-finite verbs, voice suffixes (causative,

passive, reflexive, reciprocal suffixes), subordinating suffixes, negative, tense aspect and modality markers of 1 and 2 on Table 2.5. Also nominal inflectional suffixes can be attached to non-finite verbs.

Suffix	**Type**	**Example**
Number	**Plural**	araba(lar), ev(ler)
Possessive	**First person singular**	araba(m), ev(im)
	Second person singular	araba(n), ev(in)
	Third person singular	araba(sı), ev(i)
	First person plural	araba(mız), ev(imiz)
	Second person plural	araba(nız), ev(iniz)
	Third person plural	araba(ları), ev(leri)
Case	**Accusative**	okul(u), ev(i)
	Dative	okul(a), ev(e)
	Locative	okul(da), ev(de)
	Ablative	okul(dan), ev(den)
	Genitive	okul(un), ben(im)

Table 2.3: Nominal inflectional suffixes

Suffix	**Type**	**Example**
Voice	**Causative**	taş(ır)mak, uyu(t)mak
	Passive	boz(ul)mak, tara(n)mak
	Reflexive	ört(ün)mek, giy(in)mek
	Reciprocal	döv(üş)mek
Negative		yap(ma)mak, et(me)mek
Copular		sürünmekte(ymiş), konuşuyor(sa)
Person		yapa(yım), okulda(sın)

Table 2.4: Voice, negative, copular and person markers

1	2	3	4	5
-(y)A (possibility)	-(y)Abil (possibility)	-DI (perfective)	-(y)DI (past copula)	-DIr (generalizing modality
	-(y)Iver (non-premaditative)	-mIş (perfective/evidential)	-(y)mIş (evidential copula)	
	-(y)Agel	-sA (conditional)	-(y)sA (conditional copula)	
	-(y)Ayaz	-(A/I)r/-z (aorist)		
	-(y)Akal	-(y)AcAK (future)		
	-(y)Adur	-(I)yor (imperfective)		
		-mAlI (obligative)		
		-mAktA (imperfective)		
		-(y)A (optative)		

Table 2.5: Tense, aspect, modality suffixes and their order when co-occured

2.2 Turkish Phonology

In Turkish suffixes' have a number of surface forms. Every suffix's structure changes according to both stem's consonants and vowels(Table 2.6 and Table

	Back		Front	
	Unrounded	**Rounded**	**Unrounded**	**Rounded**
Open	a	o	e	ö
Close	ı	u	i	ü

Table 2.6: Table of vowel categorization of the Turkish Language

2.8). For instance, plural suffix has two forms according to stem's last vowel's type. If the stem's last vowel's type is back, a, ı, o, u, plural suffix has surface form of -lar and if the stem's last vowel's type is back, e, i, ö, ü, plural suffix have surface form of -ler. Table 2.7 gives an example for this. This alternation on vowels of the suffixes are the result of the vowel harmony, which is related to phonological properties of the phonemes, and to keep vowel harmony vowels of the suffixes changes.

stem	**last vowel**	**vowel type**	**suffix**	**final**
araba	a	back	-lar	arabalar
ev	e	front	-lar	evler

Table 2.7: Example of the change in suffix's surface form according to category of the last vowel of the stem

Voiced	**Voiceless**
b, c, d, g, ğ, j, l, m, n, r, v, y, z	ç, f, h, k, s, ş, p, t

Table 2.8: Voice categorization of the Turkish consonants

Another change on surface forms of suffixes occur on suffixes which starts with certain consonants. If a suffix starts with a voiceless consonants ç, t, k or voiced consonants c, d, g, these consonants change to their voiceless-voiced counterparts in certain situations. If a stem ends with a voiceless consonant, suffixes, those satisfy the condition stated above, will change their first consonant with voiceless counterpart and otherwise will change their first consonant with voiced counterpart. For instance, "savaş-çı" and "diz-gi". This happens beause of to keep the consınant harmony.

If we want to list suffixation alternations other than harmonic ones there are two of them. The first one is first vowel or first consonant of a suffix that can be removed under a condition. The condition is, in Turkish vowels can only be

word	suffix	final
ben	-a	bana
sen	-a	sana
karmak	-r	karar
taramak	-r	tarar
sen	-ca	sence
para	-ca	paranca
o	-lar	onlar

Table 2.9: Examples for the irregular changes in stems and suffixes

followed by a consonant. So if a suffix, that starts with a vowel, attached to a stem, that ends with a vowel, lose its first vowel in order the maintain the mentioned rule. For instance, “kar-ar” and “tara-r”. The second one is, like in English, about irregularity and examples are given in Table 2.9 . Pronouns “ben” and “sen” become “bana” and “sana” when dattive suffix is attached or when a plural suffix, case suffix, adverbial suffix or adjectival suffix attached to some stems like “para” consonant “n” will appear between stem and suffix. So when adverbial suffix “-ca” attached to stem “para” it becomes “para-n-ca” instead of “para-ca”.

Type	**Nominal**				**Verb**	**Postposition**	**Conjunction**	**Interjection**
	Noun	**Pronoun**	**Adjective**	**Adverb**				
Word	Kitap	Ben	Çalışkan	Çok	Kızmak	Gibi	Ve	Hey

Table 2.10: Examples for Turkish word classes

Last thing we can mention about Turkish Morphology is word classes. There are five word classes in Turkish which are nominal, verb, postposition, conjunction and interjection. Also for nominal word class there are four different type which are noun, pronoun, adjective and adverb. Table 2.10 contains a list of world classes with examples for each one.

Chapter 3

Literature Survey

3.1 General Chunking and Dependency Parsing

In NLP, chunks are the next semantic group in the sentence structure after the words. Chunks are groups of words that form a coherent whole syntactically and semantically. For instance, when we read up a name we do not seperate the name and surname, title and name, or an object and its property as they are semantically seperate elements in a sentence. Instead, we band them together to make them semantically correct like in while distinguishing "third house" from just "third" and "house" we need both object and its property because as a complementary elements their meanings are ambiguous (like "third of what?", "which house?") and this process is named as chunking in NLP.

Chunking can be used in Named Entity Recognition (NER) problem [32] to enhance the results or can be used to enchance the extraction of clauses from texts to form a merged story [23]. From this perspective chunking is an important area in NLP to preprocess the data to help the other areas those benefit from the NLP such as information retrieval and machine translation.

The problem of the chunking is to determine the boundaries and types of the chunks. To determine the chunk boundaries and the types there is a need of different types of informations those extracted from words. These clues include but are not limited to morphological analysis, part of speech of the word and

its neighboors' and location of the word in the sentence. Example of a sentence chunking is in Table 3.1 .

The little yellow dog	barked	at	the cat

Table 3.1: Example of chunking of a sentence. Each column represents a chunk

To solve this problem there are different approaches. These approaches can be learning based [18], rules based [10] or both models used together [27]. The difference between approaches is the result of the structural differences between the languages which are the subject of the chunking. Determining a chunk boundary and a chunk type in languages like English is relatively easy compared to free order language considered for the same task. The reason for this is, in English, word order, which is subject, verb and object order (SOV), mostly remains same with some exceptions when indirect objects or adverbials used.

Unlike English language, laguages with free word order, like Turkish, Korean and Japanese, can not benefit much from the positional information of the word and its neighboors' to determine the chunk type [27]. For languages with free word order, common approach is using all the words in a varying sized window [7] and/or using postpositions [27]. Using contextual information around a word can help to enchance the performance but even for languages like English, at some point, increasing the window size results in poor accuracy because of the processing of unnecassary information, noise, that is captured in the window[9].

In NLP there are two types of grammars, constituency grammar and dependency grammar. In constituency grammar, it is assumed that, a sentence is a collection of constituents like noun phrases (NPs), and verb phrases (VP). Also this constituents can be nested, so one element in a sentence can be represented by one or more constituents and at the end a sentence is a combination of NPs, subject and VPs. In dependency grammar, links occur between the elements, words (instead of constituents), of the sentence. Every word can be linked to many word, with a restriction, by a directed link and at the end every word of the sentence directly or indirectly linked to verb. These directed links between the word pairs are called

dependency and dependecies also attach an information to the related word pairs which is called head-dependent (or tail). The restriction we have mentioned for link amount of a word is about the number of head and dependent a word can have. Each word can have many dependents but have only one head. Comparison of the mentioned grammars' parse trees is in Figure 3.1 .

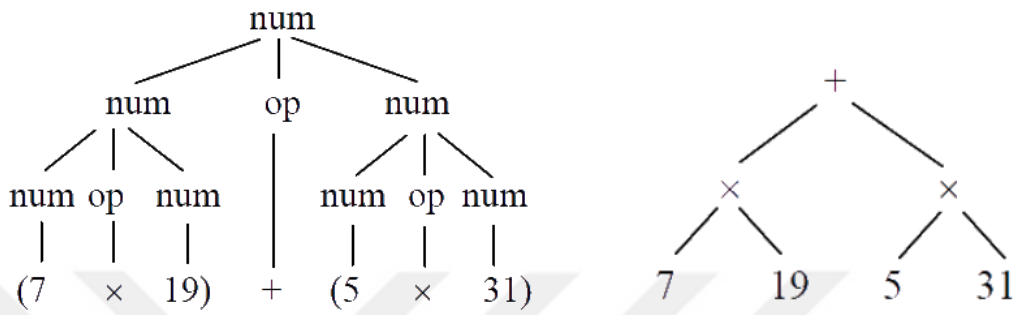

Figure 3.1: Kahane's comparison of constituency tree and dependency tree of the same mathematical equation. Parse tree at the left is constituency tree while right one is dependency tree.

In any word pair, head is the word that specifies the necessity, type and form of the dependent(s). Because of this, it is possible to guess dependent(s) for a given head but not vice versa. Common used direction of dependency relation links is head to dependent. Also every dependency can be specialized by giving a class of relation between head and dependent to supply further information or even further, simplify the dependency tree while supplying further information. Head-dependent relation is shown in Figure 3.2 .

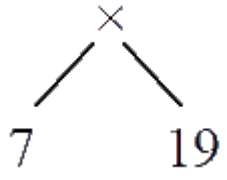

Figure 3.2: Example of head-dependent relation in dependency tree. Product (x) is the head of 2 numbers (dependents) of the operation.

Ibn Mada, was a linguist who lived in twelfth century, may have been the first person who come up with the dependencies between the words like in modern

dependency grammar for Arabic [8]. Also there are works on dependency grammar in nineteenth century. The beginning of the modern dependency grammar is based on the work of French linguist Tesniere. Tesniere represented the sentences as a tree like diagrams which he called stemmas [31]. In Tesniere's work stemmas do not follow the actual word order of the sentence, instead stemmas are structured as a hierarchical representation of the sentence. Tesniere did this seperation because he thought when sentences are created, the speaker or the listener do not actually follow the word order to create the relations between words to understand the actual meaning. So a person, unintentionally, converts between the actual and hierarchical word order in the process of communication.

S. Kahane [16] discussed why to choose dependency grammar rather than constituency grammar. He pointed that a dependency tree can contain more information, semantically, while being simpler than constituency tree because it requires information about the heads to be added to considiration to fully create a dependency tree. He also compared, simple to complex, different structures of constituency and dependency trees like headed and co-headed constituency trees and stratified and bubble dependency trees and their equivalency (Figure 3.3). The last thing he mentioned to choose dependency grammar is correctness in representation of the syntactic structure of a language because of the current models can not handle the complexity of the syntactic structures.

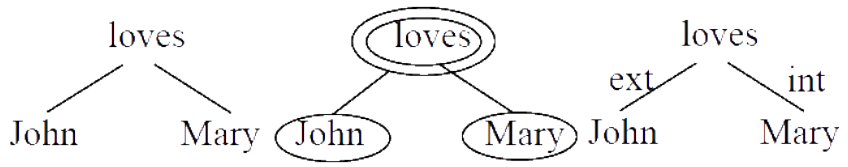

Figure 3.3: Different types of dependency trees. First one is simple dependency tree that does not contain exact head information while second one has head information and third one also specifies the relation types.

3.2 Semantic Role Labelling

Finding out semantic relation among the predicate in a sentence and its arguments is called semantic role labelling. Semantic labelling is done according to a set of predefined semantic roles like Agent or Force, volitional / non-volitional causer of the event, Theme, most directly affected by the event [15] but basically, this is a task to answer the questions "who" did "what" to "whom", "where", "when" and "how" [22] (Table 3.2). SRL can enchance the result of machine question answering [30] or machine translation [20] .

When	**Who**	**What**	**How**	**Where**	**Whom**
Today	the little yellow dog	barked	loudly	from the car	to a man

Table 3.2: Example of chunking based on "who" did "what" to "whom", "where", "when" and "how"

PropBank, NomBank and FrameNet are the projects to create a data which are annotated with semantic roles. PropBank data is created by annotation of the Penn TreeBank data by P. Kingsbury [17] , according to semantic roles they have created for each verb. The main focus of PropBank is verb predicates. NomBank is related to PropBank but for noun predicates. FrameNet is also related to PropBank but the difference is while PropBank specifies the semantic role of each single verb, FrameNet specifies the semantic role of a frame which is a collection of semantically related words like football, ball, refree, stadium, field, ticket.

The main approach to SRL is based on supervised machine learning by using the annotated data from the projects like PropBank, NomBank [14] and FrameBank.

3.3 Turkish Dependency Parsing

Turkish commonly follow subject-object-verb (SOV) order in written texts but as mentioned before Turkish is a free word order, agglutinative language and each of the six variants of the SOV can be seen in word order according to context

([11] , [7]). So words in a sentence can move freely across the sentence under some constraints and Table 3.3 has an example for this.

Ahmet'in dedesinden kalan büyük bir evi vardı.
Ahmet'in büyük bir evi vardı dedesinden kalan.
Büyük bir evi vardı Ahmet'in dedesinden kalan.
Dedesinden kalan büyük bir evi vardı Ahmet'in.

Table 3.3: Example of Turkish free order sentence. All of the 4 sentences have the same basic meaning. Also it is possible to create more variations.

K. Oflazer [25] applied extended finite state machine to Turkish Dependency Parsing which did not performed well when used on sentences with high number of verbal adjuncts as modifiers.

Eryiğit et al. [7] used inflectional groups (IG) to find dependency relations between words. They assumed the last IG determines the role of the word as a dependent and links to one of the IGs in the head word. The reason they choose to use IGs is that they assumed each IG require different kind of dependents and all of the IGs, instead of just whole word, need to be considered to form dependency relations between words. Their statistical approach achieved 73.5% accuracy on IG to IG links by just looking last IGs of the words which come before and after the dependent. Figure 3.4 contains an example of their IG based approach.

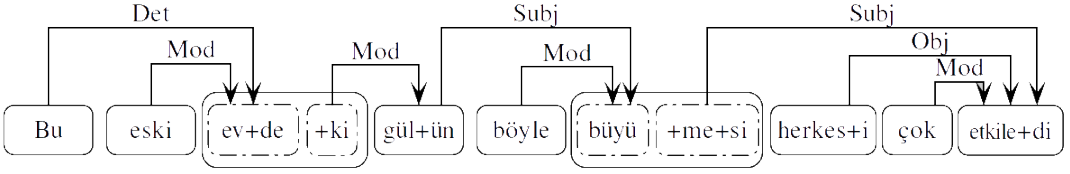

Figure 3.4: Inflectional group (IG) based dependency example, Eryiğit et al., 2006.

Nivre et al [24] used support vector machine (SVM) to parse Swedish and Turkish which improved the result of the previous statistical approach ([7]).

Çakıcı et al. [3] compared the performance of the Collins model with maximum spanning tree model on four different tag sets and for each tag set there are two

origin, from tagger or golden tags. They reported that simple extansions on tag set improves accuracy on all used models.

3.4 Turkish Semantic Role Labelling

İşgüder-Şahin et al. [12] showed that mophosemantics of the morphologically rich languages can be used to improve performance on many areas related to semantics including SRL. They exemplified this with some comparison between few Turkish sentences with English ones which contain the destination and starting location information. They showed that while English sentences can be composed of many different prepositions to indicate the word is destination or starting location, in Turkish, this is achieved by using dative and ablative case respectively. According to their observations, they suggested that case markings in Turkish can be useful for Turkish SRL.

İşgüder et al. [13] achieved 90% accuracy on determining verb predicates and their arguments by using morphological features and dependency parse trees of the sentences. They assumed all forms of verbs like verb-noun ("gitmek"), verb-adjective ("Görülecek iş"), as a perdicate. For determining the arguments they first looked to the parent of the predicate and if the parent has one child and is noun or adjective, it is selected as parent argument otherwise they looked to the each child of the predicate and if the child is not a single root or coordination node, it is selected as child argument. After that they decide the coordination arguments if there are more than one predicate that use same argument. One of the predicate they have labeled correctly is shown in Table 3.4 .

Ama ben bütün olaylara pozitif **bakıyorum**

Table 3.4: İşgüder-Şahin et al., 2014, example of predicate

Chapter 4

Approach

There are only four results on dependency parsing of Turkish and the main approach to the area is using neighboring inflectional groups to determine dependencies ([7]). In their work, they have used combination of 1 unit left and/or 1 unit right of the dependent and/or the head to decide on dependency relation and achieved 3% more accuracy than their baseline parser.

Unlike the mentioned approach above we have proposed a new approach to decide where to look for a dependency relations. Semantic roles are, after the words, the smallest semantic units of a sentence. From this information, we assumed that dependency relations only occur between the elements of a semantic role and between those semantic roles.

Also there is no tool or research to follow on Turkish SRL that satisfy all of the semantic roles. For this reason we have labelled the data with five different semantic roles according to our rule based steps. After the SRL we formed dependencies between the elements. Thus, we divided the task into two sub-tasks. SRL at the sentence level and dependency at the role level.

Our approach consists of 4 different steps: data creation, rule based semantic boundary finding, labelling semantic chunks and creating dependency relations. In first step, we processed the data to make it useful for our need by extracting morphological features. Then, using a tool that we created, we labelled the

boundaries. In second step, we run two different rule sets on data. In first rule set, we have created series of rules which use morphological features, POS tags of current word and surrounding words to determine whether there is a boundary or not after the current word. In second rule set we have created a series of rules that use boundary informations additionally which we have created by running the first rule set to determine the remaining boundary informations. In third step, we have created small set of rules to label the chunks with semantic roles of direct object, indirect object, adjunct, subject and predicate. In fourth step, we have created the dependencies between the elements.

Thus, in our approach, chunks corresponds to units that are semantic roles. We claim this is a better splitting of the sentence at shallow parsing level. For example, in sentence "büyük evin küçük kapısı kırıldı" we could identify 'büyük ev' and 'küçük kapı' as NP chunks at the shallow level. However, the whole chunk stands as a Genitive-Possessive unit and should be treated as a whole in terms of relations at the sentence level. Once this sequence is identified, its internal dependencies can be decided without any clues from outside the sequence.

Our second man contribution is the two-stage identification of chunks. In the first stage, we identify sentence level roles and in the second stage, we identify chunk level dependencies.

We also separate chunk boundary identification from their labelling. Labelling needs to use only the boundary attachment elements such as case markings at the boundary. Boundary detection, on the other hand, makes use of all the contextual information that can be extracted at the word and sentence levels.

Chapter 5

Data Preparation

5.1 Corpus

We have used modified METU-Sabancı Turkish Treebank for universal dependencies. There are 3948 Conll-u formated sentences in the corpus ([6] , [26] , [2]). We have extracted the sentences from the corpus and than by using a morphological analyzer [28] [5] and a disambiguator [5] we have extracted the morphological information from the data and saved it together in Conll format. We have removed some of the sentences and used total of 3882 sentences.

5.2 Modifications

We have modified the selected corpus in order to make it fit our need. We have added two new tags, <SB> and <SE>, to indicate sentence boundaries in the data instead of using numeration and empty lines. Our created data have three columns. First column contains the original word or the sentence boundary information, second column contains the root of the word on the first column, third column contains the disambiguated morphologic analysis and POS tags of the transformation of the root on the second column to become the word on the first column.

5.3 Morphological Analysis

We have used morphological analyzer and morphological disambiguator from the ITU Turkish Natural Language Pipeline [5] . The output of the morphological analyzer contains all of the possible morphological structures of the given word and the disambiguator selects the correct one according to the sentence's flow like shown in Table 5.1.

Word	**Morphological Analysis (Disambiguated)**
Ama	**(ama+Conj)** ama+Adj ama+Noun+NAdj+A3sg+Pnon+Nom am+Noun+A3sg+Pnon+Dat
annemin	**(anne+Noun+A3sg+P1sg+Gen)**
şartları	**(şart+Noun+A3pl+Pnon+Acc)** şart+Noun+A3pl+P3sg+Nom şart+Noun+A3sg+P3pl+Nom
vardı	**(var+Noun+NAdj+A3sg+Pnon+Nomˆ DB+Verb+Zero+Past+A3sg)** var+Verb+Pos+Past+A3sg
.	**(.+Punc)**

Table 5.1: Example output of "Ama annemin şartları vardı ." sentence after morphologicaly analyzed and disambiguated. Selected analysis by disambiguator is written in bold and in paranthesis.

5.4 Tool

After the morphological data extracted, we have created rule based semantic boundaries over data with the extracted information. Also to compare our result with gold standard data, we had to create manually labelled data. To do that, first, we created an easy tool to label the data. Tool just basically traverses the files which follow the naming convention of "chnkXXXXX.txt" and present the each sentence visualy to tagger. After that tagger can create, expand, shrink or remove semantic chunks, thus semantic boundaries for each word, for comparison and also can label those chunks to use in next steps.

Tool uses special data format to store he information. Data will be kept between <DATA>, </DATA> tags in two columns. Each line between <DATA>, </DATA> will contain the a word from the sentence in the first column and the label in the second column. BI notation used for labels to indicate where the label starts and ends. Between the <META> and </META> tags, tool stores

the last modified time, tagger ID and notes for the each chunk if presents. Tool's data format is presented in Table 5.2.

<DATA>
Ama B-ADJ
annemin B-D_ OBJ
şartları I-D_ OBJ
vardı B-PRED
.
</DATA>
<META>
lastModified:
taggerID:
note:
</META>

Table 5.2: Example of data structure for the tool

5.5 Manual Labelling

By using our tagging tool, a tagger can easily traverse over sentences and can create, expand, shrink or remove semantic chunks.

To create a chunk, a tagger can just click on a word to create a chunk that contains the clicked word or just draw a rectangle by left clicking and dragging at the same time to add all the words those under the rectangle to create a chunk that contains them when the mouse button released.

To expand a chunk, a tagger can just left click on the buttons that placed at either side of the chunk to expand the chunk on that direction. By doing this chunk includes, if presents, the word on the desired direction. If the word that is going to be added to the current chunk is a part of an other chunk, it got removed from the other chunk, in other way the latter is shrinked.

To shrink a chunk, a tagger can just draw a rectangle by right clicking and dragging at the same time on the chunk that is wanted to be shrink. When the mouse button released the words those are under the rectangle will be removed

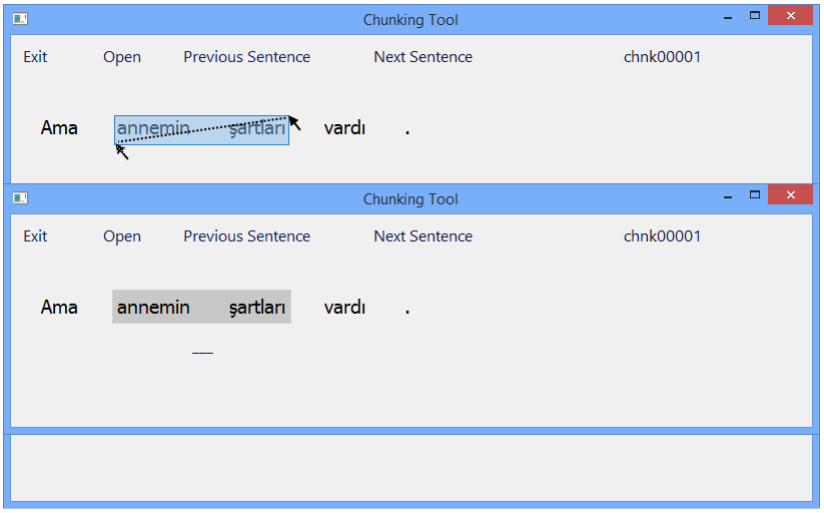

Figure 5.1: Example of chunk creation.

from the chunk and the remaining part of the chunk will keep it's words and, if presents, it's label.

To remove a chunk, a tagger can just select any chunk by left clicking on it and pressing "delete" key on the keyboard or like in shrinking can draw a rectangle that overlaps the whole chunk.

By using our tool we have tagged a total of 1000 sentences for chunk boundaries and labeled the chunks of the 170 of them as a test and training sets.

5.6 Semantic Chunks

We have used five different semantic chunk types, Predicate, Subject, Direct object, Indirect object and Adjunct. While we have maintained the universal definitions of the roles of direct/indirect objects and subject, we have made some modifications on the definitions of the roles of predicate and adjunct.

5.6.1 Subject

The subject is who or what is doing or being something in he sentence. In Turkish the subject is not always overtly expressed within the sentence [9] , instead,

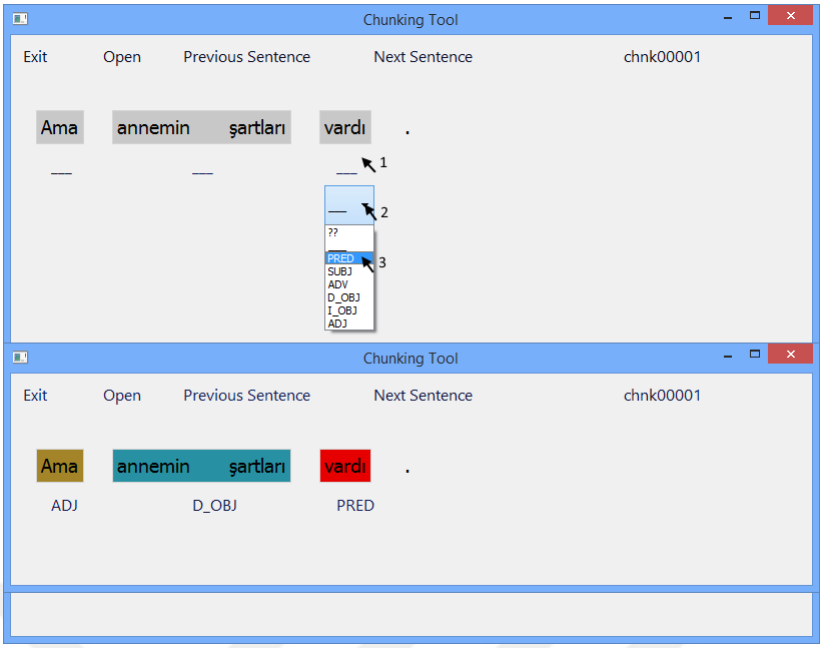

Figure 5.2: Example of how to label a semantic chunk and final form of the sentence after semantically labeled.

sometimes the subject information can only be extracted from markings of the predicates or from the markings of the adverbials, in our work included in adjunct.

Ahmet	derslere	gelmemeye	başlamıştı	.

Table 5.3: Exmaple of subject of a sentence

5.6.2 Direct Object

Direct object is the person or the thing which is brought into being or which something is done by the action of the subject [9] .

Benden sonra aşık olduğu adamı	gece	gündüz	izledim	.

Table 5.4: Example of direct object of a sentence

5.6.3 Indirect Object

Indirect object is the person or the thing that does not directly affected by the subject's action but also is a part of the meaning.

Her	hafta	bir kitap	okurum	.

Table 5.5: Example of indirect object of a sentence

5.6.4 Predicate

Predicate is an event, a process or a state that the subject is involved. There are two types of sentences in Turkish according to their predicate's types, verbal if it includes verb predicate and nominal if it includes noun predicate. In our work we have kept the universal structure of the verb predicate but included everything that directly modifies the predicate in nominal sentences to noun predicate.

Sonunda	başka	bir	yol	buldular	.

Table 5.6: Example of predicate of a sentence

5.6.5 Adjunct

Additional to common use of adjuncts we have labelled everything left out in the sentence as an adjunct. These includes location adverbials, time adverbials etc. .

Şaşkın şaşkın	onları	izledim	.

Table 5.7: Example of adjunct of a sentence

Chapter 6

Rules

We have used 3 different rule sets on our data. Two of them are used to determine the semantic role boundaries and the last one is used to determine the type of the semantic role. Under this section, we will give examples for the each rule we have created. For our first rule set we have created 81 rules to determine semantic role boundaries from morphological clues and the POS tags of the current and surrounding words. Rules are weighted We used Bound tag if there is a boundary after the word or non-Bound if there is none. Rule with their examples can be seen in Table 6.1 , Table 6.2 , Table 6.3

Rule Number	**Description** / **Example**		**Rule Weight**	**Bound** / **non-Bound**
1.	**If the current word has Adj feature**		**0.1**	**non-Bound**
	Sanıyorum bütün entellektüel züppeler oradalar .			
	Bütün	*Adj*		*non-Bound*
	entellektüel	*Adj*		*non-Bound*
	züppeler	Noun NAdj A3pl Pnon Nom		Bound
2.	**If the next word's POS tag is Conj**		**0,1**	**non-Bound**
	Adana'ya nakil girişimlerinden de haberliyim			
	girişimlerinden	Noun A3pl P3sg Abl		*non-Bound*
	de	*Conj*		Bound
3.	**If the current word's POS tag is Postp**		**1**	**Bound**
	... bir gecelik gösteri için gelmiş .			
	için	*Postp PcNom*		*Bound*
	gelmiş	Verb Pos Narr A3sg		Bound
4.	**If the current word has Loc feature and the next word's POS tag is not Verb**		**1**	**non-Bound**
	... ilişkin bilginin sözleşmede yer alması ...			

Table 6.1 continued from previous page

Rule Number	Description / Example		Rule Weight	Bound / non-Bound
	sözleşmede	*Noun A3sg Pnon Loc*		*non-Bound*
	yer	*Noun A3sg Pnon Nom*		non-Bound
5.	**If the current word has NAdj feature and the next word's POS tag is not Verb**		**1**	**non-Bound**
	Erkekler Parkı'na gidiyorsun .			
	Erkekler	*Noun NAdj A3pl Pnon Nom*		*non-Bound*
	Parkı'na	*Noun A3sg P3sg Dat*		Bound
6.	**If the current word has P3sg, Acc features**			**Bound**
	Kız sabah olunca rüyasını hatırlamış .			
	rüyasını	*Noun A3sg P3sg Acc*		*Bound*
7.	**If the current word has Pnon, Nom features and the next word has P3sg, Nom features**		**1**	**non-Bound**
	İnce ayar söz konusu .			
	söz	*Noun A3sg Pnon Nom*		*non-Bound*
	konusu	*Noun A3sg P3sg Nom*		Bound

Table 6.1 continued from previous page

Rule Number	Description Example		Rule Weight	Bound non-Bound
8.	**If the current word's POS tag is Adverb**		**0.1**	**non-Bound**
	... fikir değiştirip dönmeye karar verdi .			
	değiştirip	*Verb Caus Pos ĴDB Adverb AfterDoingSo*		*non-Bound*
	dönmeye	Verb Pos ĴDB Noun Inf2 A3sg Pnon Dat		Bound
9.	**If the current word has Demons feature and the next word's has Pers or Ques feature**		**1**	**non-Bound**
	Bütün bunlara rağmen ...			
	bunlara	*Pron Demons A3pl Pnon Dat*		*non-Bound*
	rağmen	*Postp PCDat*		Bound
10.	**If the current word's POS tag is Noun and the next word's POS tag is Postp and it has PCNom feature**		**1**	**non-Bound**
	Saçları aslan yelesi gibi kabarıktı .			
	yelesi	*Noun A3sg P3sg Nom*		*non-Bound*
	gibi	*Postp PCNom*		Bound

Table 6.1 continued from previous page

Rule Number	Description / Example		Rule Weight	Bound / non-Bound
11.	**If the current word has Acc feature and the next word has xxxPart feature**		**1**	**non-Bound**
	... İstanbul'da kalmayı planlayan genç kadın ...			
	kalmayı	*Verb Pos ĴDB Noun Inf2 A3sg Pnon Acc*		*non-Bound*
	planlayan	*Verb Pos ĴDB Adj PresPart*		non-Bound
12.	**If the current word has Quant feature**		**1**	**Bound**
	Ordakilerin hepsi senin gibi .			
	hepsi	*Pron Quant A3pl P3pl Nom*		*Bound*
13.	**If the current word's POS tag is Postp and has Ques feature**		**1**	**Bound**
	... anaların yatıştırmalarıyla sakinleşmez mi ortalık .			
	mi	*Postp Ques Pres A3sg*		*Bound*
14.	**If the current word's root and the next one's root are same**		**1**	**non-Bound**
	Ömür Uzatma Kıraathanesi'nin kapısı güm güm vuruldu .			
	güm	*güm*		*non-Bound*
	güm	*güm*		Bound

Table 6.1 continued from previous page

Rule Number	**Description** / **Example**		**Rule Weight**	**Bound** / **non-Bound**
15.	**If the current word's root and the previous one's root are same**		**1**	**Bound**
	Ömür Uzatma Kıraathanesi'nin kapısı güm güm vuruldu .			
	güm	*güm*		non-Bound
	güm	*güm*		*Bound*

TABLE 6.1: 15 most effective rules from the first rule set with their output when applied to data. Italic lines indicates parts of the example related to rule

Rule Number	**Description** / **Example**		**Bound** / **non-Bound**
1.	**If there is no boundary (non-Bound) after previous and next words and the current word has Pnon, Nom features**		**non-Bound**
	Dolaştığın o itler var ya ...		
	o		*non-Bound*
	itler	*Noun A3pl Pnon Nom*	*non-Bound*
	var		*non-Bound*
2.	**If there is no boundary (non-Bound) after previous word and boundary (Bound) after next word and the current word has Pnon, Nom features**		**Bound**
	... Darwin'le bile barışmakta bir beis görmemiştir .		
	bir		*non-Bound*
	beis	*Noun A3sg Pnon Nom*	*Bound*
	görmemiştir		*Bound*
3.	**If there is no boundary (non-Bound) after previous and next words, next word is not the last word of the sentence and the current word has P3sg, Nom features**		**Bound**

Table 6.2 continued from previous page

Rule Number	**Description** / **Example**		**Bound** / **non-Bound**
	... hapis cezası istemiyle fezleke hazırladı .		
	hapis		*non-Bound*
	cezası	*Noun A3sg P3sg Nom*	*non-Bound*
	istemiyle		*non-Bound*
4.	**If there is no boundary (non-Bound) after previous and next words and the current word has Pnon, Gen features**		**non-Bound**
	Yüksek teknolojinin yetersiz kaldığı bu havanın ...		
	Yüksek		*non-Bound*
	teknolojinin	*Noun A3sg Pnon Gen*	*non-Bound*
	yetersiz		*non-Bound*
5.	**If there is no boundary (non-Bound) after previous and next words and the current word has Dat feature**		**non-Bound**
	... yeni yeni kendini alıştırmaya başlamış olan ...		
	kendini		*non-Bound*

Table 6.2 continued from previous page

Rule Number	Description Example		Bound non-Bound
	alıştırmaya	*Verb Caus Pos ĴDB Noun Inf2 A3sg Pnon Dat*	*non-Bound*
	başlamış		*non-Bound*
6.	**If there is no boundary (non-Bound) after previous word and boundary (Bound) after next word, next word's POS tag is not Verb and the current word has P3sg, Nom features**		**non-Bound**
	Bir de bana bir sürprizi olacağından bahsediyor .		
	bir		non-Bound
	sürprizi	*Noun A3sg P3sg Nom*	*non-Bound*
	olacağından		*Bound*
7.	**If there is boundary (Bound) after previous word and no boundary (non-Bound) after next word and the current word has Pnon, Nom features**		**non-Bound**
	... tıpkı bir çocuk gibi ben merkezci olmaları ...		
	gibi		*Bound*
	ben	*Pron Pers A1sg Pnon Nom*	*non-Bound*

Table 6.2 continued from previous page

Rule Number	Description ——— Example		Bound ——— non-Bound
	merkezci		*non-Bound*
8.	**If there is no boundary (non-Bound) after previous word and boundary (Bound) after next word and the current word has Pnon, Gen features**		**non-Bound**
	Rastgele bir odanın kapısını vurdum .		
	bir		*non-Bound*
	odanın	*Noun A3sg Pnon Gen*	*non-Bound*
	kapısını		*Bound*
9.	**If there is boundary (Bound) after previous word and no boundary (non-Bound) after next word and the current word has Dat feature**		**non-Bound**
	Ara ara aklıma takılan soruları da anlatır gibi ...		
	ara		*Bound*
	aklıma	*Noun A3sg P1sg Dat*	non-Bound
	takılan		non-Bound

Table 6.2 continued from previous page

Rule Number	Description ——— Example		Bound ——— non-Bound
10.	**If there is boundary (Bound) after the next word and next word's POS tag is Verb and the current word has Abl feature**		**Bound**
	Çünkü çocuğun dünyası saf duygudan ibarettir .		
	duygudan	*Noun A3sg Pnon Abl*	*Bound*
	ibarettir	Noun NAdj A3sg Pnon Nom ĤDB Verb Zero Pres A3sg Cop	*Bound*

TABLE 6.2: 10 most effective rules from the second rule set with their output when applied to data. Italic lines indicates parts of the example related to rule

Rule Number	**Description** / **Example**		**Label**
1.	**Examine chunks from last to first and the first chunk ends with a Verb**		**Predicate (PRED)**
	Ona her şeyimi verdim .		
	Ona		ADJ
	her şeyimi		D_ OBJ
	verdim	*verdim -> Verb Pos Past A1sg*	*PRED*
2.	**Examine chunks and if a chunk ends with a word with Acc feature**		**Direct Object (D_ OBJ)**
	Ona her şeyimi verdim .		
	Ona		ADJ
	her şeyimi	*şeyimi -> Noun A3sg P1sg Acc*	*D_ OBJ*
	verdim		PRED
3.	**Examine chunks and if a chunk ends with a word with Pron POS tag**		**Subject (SUBJ)**
	Sen o kadar anlatmışsın .		
	Sen	*sen -> Pron Pers A2sg Pnon Nom*	*SUBJ*
	o kadar		ADJ
	anlatmışsın		PRED

Table 6.3 continued from previous page

Rule Number	**Description** / **Example**		**Label**
4.	**Examine chunks and if any unlabeled chunk that has same (xx)sg or (xx)pl feature with predicate and also have Prop, Nom or Pnon, Nom features pairs**		**Subject (SUBJ)**
	Kot pantolonla gelen hiçbir çocuk yoktu .		
	Kot pantolonla gelen hiçbir çocuk	*çocuk -> Noun A3sg Pnon Nom*	*SUBJ*
	yoktu		PRED

TABLE 6.3: 4 of the semantic chunk labeling rules and their outputs when applied to data. Italic lines indicates parts of the example related to rule

Chapter 7

Evaluation

7.1 Rule Performance

As we mentioned before we have three different rule sets. Two for determining the semantic chunk boundaries and one for labelling the semantic chunks.

We have used first two rule sets on a 1000 sentences data set for boundaries which is previously tagged by the tagger and the last rule set on a 170 sentences data set for role labels which is also labeled by the tagger.

7.1.1 Determining the Semantic Chunk Boundaries

To determine the semantic boundaries we have applied the rule sets we created. Each word can fire multiple rules. So to decide whether there is a boundary after a word or not we have adopted majority voting. If boundary and no boundary rule counts are equal we have selected no boundary for the current word.

We have used the first rule set on the data that only contains the morphological features and POS tags of the words. By using this rule set we have achieved 77.5% accuracy on the boundaries that we can decide. Example output compared to actual boundaries can be seen in Table 7.1 .

We have used the second rule set on the data that contains the morphological features, POS tags and the boundary information that we extracted from the

Word	Gold	Rules Output	Fired Rules
Sanal	non-Bound	non-Bound	R11, R38, R48
parçacıklarsa	Bound	Bound	RE7
bunların	non-Bound	non-Bound	R3
hiçbirini	Bound	Bound	R24, R28
yapamazlar	Bound	Bound	R20, R70
.			

Table 7.1: Example of placing semantic boundaries on words from morphological features and POS tags. First column is word of the sentence, second column is the tagger's decision for whether there is a semantic boundary or not after the word, third one is the decision based on fourth column's fired rules.

Word	Gold	Rules Output	Fired Rules
Bir	Bound	Bound	R1, R11
sigara	Bound	Bound	BR21, BR31
yakmıştı	non-Bound	non-Bound	R70
.			

Table 7.2: Example of placing semantic boundaries on words from boundary information extracted from previous step in addition to morphological features and POS tags.

first run. By using this rule set we obtained our 77% accuracy on the boundaries that we can decide, with 11.5% increase in the number of decided boundaries. Example output compared to actual boundaries can be seen in Table 7.2 .

We have used linear chain Conditional Random Fields (CRF) which is a popular statistical modeling method used for NLP to compare its output with our results. CRF takes context into account, by traversing sequence of inputs, to produce an output unlike a simple classifier. In our work, we have used multiple input to make decision too, so using CRF for comparison suits in our case well. When we have give the features in Table 7.3 to CRF for semantic boundary creation, CRF has achieved 80% accuracy on the data. We used 10-fold cross validation on 1000 sentences, 100 sentences for test and 900 sentences for train for each. We did not used any punctuation information throughout the phases, so to maintain this, we also stripped off the punctuations, except quotation marks, from the data for CRF.

We have used the third rule set on the data that contains the morphological

Previous word
Current word
Next word
Current word's position (Start, None, End)
Next word's position (Start, None, End)
Previous word's type (Noun, Verb, etc.)
Current word's type (Noun, Verb, etc.)
Next word's type (Noun, Verb, etc.)
Current word's root is equal to next
Current word's root is equal to previous
Previous word has DoingSo
Currentcurrent word has DoingSo
Next word has DoingSo
Next word has Gen, Nom, None
Previous word has Acc
Current word has Acc
Next word has Acc
Previous word has Loc
Current word has Loc
Next word has Loc
Previous word has Axyy or Pxyy
Current word has Axyy or Pxyy
Next word has Axyy or Pxyy
Current word has Pers
Current word's tense

Table 7.3: Features used in CFR.

features, POS tags and the boundary informations that we extracted from the previous runs. By using this rule set we have achieved 82.5% accuracy on the labels of the semantic chunks. Example output compared to actual semantic roles can be seen in Table 7.4 .

Word	**Gold**	**Rules Output**
Sanal	ADJ	ADJ
parçacıklarsa		
bunların	Direct Object	Direct Object
hiçbirini		
yapamazlar	Predicate	Predicate
.		

Table 7.4: Example of labeling semantic chunk with roles.

7.2 Rule Selection

To decide to which morphological features we will use in our rules to decide semantic boundaries, we have examined the relations of the words' morphological features. For this examination, we have created uni-gram (Table 7.5) and bi-gram (Table 7.6) relations of the morphological features and then from the relations and the examination of the related data, we have selected which feature will have higher impact on accuracy. After the rule creation we have removed the rules, which act like other rules' sub-rules, which have zero effect on accuracy.

POS Tag + Features	**Number of Occurence**	**Frequency**
Noun+A3sg+Pnon+Nom	3494	11.15%
Adj	2423	7.74%
Adverb	1617	5.16%
Conj	1439	4.60%
Adj+Num	1171	3.74%
Noun+A3sg+P3sg+Nom	807	2.58%
Noun+Prop+A3sg+Pnon+Nom	786	2.51%
Det	784	2.50%
Verb+Pos+Past+A3sg	652	2.08%
Adj+PresPart	601	1.92%

Table 7.5: Top 10 uni-gram POS tag + features of the words in the data set.

POS Tag + Features	**Pos Tag + Features**	**Number of Occurence**	**Frequency**
Adj+Num	Noun+A3sg+Pnon+Nom	538	1.94%
Adj	Noun+A3sg+Pnon+Nom	332	1.20%
Noun+A3sg+Pnon+Nom	Noun+A3sg+P3sg+Nom	313	1.13%
Adj	Adj+Num	240	0.86%
Noun+A3sg+Pnon+Nom	Noun+A3sg+Pnon+Nom	234	0.84%
Adverb	Adj	229	0.82%
Det	Noun+A3sg+Pnon+Nom	210	0.76%
Noun+Prop+A3sg+Pnon+Nom	Noun+Prop+A3sg+Pnon+Nom	170	0.61%
Conj	Adj	158	0.57%
Adj	Adj	158	0.57%

Table 7.6: Top 10 bi-gram POS tag + features of the words in the data set.

Chapter 8

Discussion

Parsing of free word order languages is a diffucult task. In these languages, location knowledge is missing when compared to languages like English, where sentence structure is somewhat fixed. To compansate this, works on Turkish assumed the SOV word order. Recent works on Turkish dependency parsing rely on IGs' relations which are assumed to be placed closely to each other.

In our work, instead of using the common approach to Turkish dependency parsing, we tried to find semantic roles in a sentence and then used this information to figure out dependency relations. Also we used rule based approach to figure out semantic boundaries and roles.

Because we have determined whether there is a semantic boundary or not after a word by looking to the morphologic structure of current and surrounding words, our rule based approach to the problem depend on the morphological analysis of the words. The morphological disambiguator that we used was not accurate on proper names and sometimes selects the wrong analysis for output. Also for some adjectives morphological analyzer outputs only noun analyses. This impacts the accuracy of our approach.

Also the data we used includes scrambled sentences, which creates a confusion to decide on boundary. This happens because of the morphological structure of the new word at the position of replaced one can have same or similar structure. To

avoid this confusion we did not create rules which corresponds to such situation. Also we did not used punctuation informations to decide on boundaries. Because of this some of the words' boundaries before or after the punctuations remain undecided.

Chapter 9

Conclusion

The problem of the dependency parsing is figuring out the sentence structure in a language by using relations between the words. These relations are hard to find in languages with free word order. For Turkish there is only a small dependency data set to work with and this impacts the learning approaches. For this reason, we have chosen the rule based approach instead of learning approach.

In this thesis, instead of directly figuring out dependency relations between words, we thought that finding semantic roles in a sentence can give us some clues to do that. This approach also eliminates the problem of where look for a dependent if semantic roles are found properly. For finding semantic roles we have used morphological analysis of the words to decide on semantic boundaries.

As a result, our work on dependency parsing is different from the others. We worked on an approach that can improve the figuring out the dependency relations in a sentence.

In the future, we plan to repeat the process with a better morphological analyzer and also find out the dependency relations inside semantic roles.

References

[1] Kübra Adalı and A. Cüneyd Tantuğ. A rule based noun phrase chunker for Turkish. *TÜRKİYE BİLİŞİM VAKFI BİLGİSAYAR BİLİMLERİ ve MÜHENDİSLİĞİ DERGİSİ*, 7(1(Basılı 8), 2015.

[2] Nart Atalay, Kemal Oflazer, and Bilge Say. The annotation process in the Turkish Treebank. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora - LINC*, 3 2003.

[3] Ruket Cakıcı and Jason Baldridge. Projective and non-projective Turkish parsing. 2006.

[4] Michael Collins. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, ACL '98, pages 16–23, Stroudsburg, PA, USA, 1997. Association for Computational Linguistics.

[5] Gülşen Eryiğit. ITU Turkish NLP web service. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, Gothenburg, Sweden, April 2014. Association for Computational Linguistics.

[6] Gulsen Eryigit, Tugay Ilbay, and Ozan Arkan Can. Multiword expressions in statistical dependency parsing. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages (IWPT - 12th International Conference on Parsing Technologies)*, pages 45–55, Dublin, Ireland, October 2011. Association for Computational Linguistics.

[7] Gülşen Eryiğit and Kemal Oflazer. Statistical dependency parsing of Turkish. In *Proceedings of the 11th EACL*, pages 89–96, Trento, 3-7 April 2006.

[8] Kim Gerdes, Eva Hajicová, and Leo Wanner, editors. *Computational Dependency Theory [papers from the International Conference on Dependency Linguistics, Depling 2011, Barcelona, Spain, September 2011]*, volume 258 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2013.

[9] A. Göksel and C. Kerslake. *Turkish: A Comprehensive Grammar*. Comprehensive grammars. Routledge, 2005.

[10] Claire Grover and Richard Tobin. Rule-based chunking and reusability. In *In Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC*, 2006.

[11] Beryl Hoffman. Integrating "free" word order syntax and information structure. In *Proceedings of the Seventh Conference on European Chapter of the Association for Computational Linguistics*, EACL '95, pages 245–252, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[12] Gözde Gül İşgüder and Eşref Adalı. Using morphosemantic information in construction of a pilot lexical semantic resource for Turkish. In *Proceedings of Workshop on Lexical and Grammatical Resources for Language Processing*, pages 46–54, 2014.

[13] Gözde Gül İşgüder and Eşref Adalı. Verb predicate and argument extraction for Turkish. 2014.

[14] Richard Johansson and Pierre Nugues. Dependency-based syntactic–semantic analysis with PropBank and NomBank. In *CoNLL 2008: Proceedings of the Twelfth Conference on Natural Language Learning*, pages 183–187, Manchester, United Kingdom, 2008.

[15] Daniel Jurafsky and James H. Martin. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2009.

[16] S. Kahane. *Why to choose dependency rather than constituency for syntax: a formal point of view.* Moscow, 2012.

[17] Paul Kingsbury and Martha Palmer. From treebank to propbank. In *LREC*. European Language Resources Association, 2002.

[18] Taku Kudo and Yuji Matsumoto. Chunking with support vector machines. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, NAACL '01, pages 1–8, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics.

[19] Rochelle Lieber. *Introducing Morphology.* Cambridge Introductions to Language and Linguistics. Cambridge University Press, 2009.

[20] Ding Liu and Daniel Gildea. Semantic role features for machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 716–724, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[21] Mitchell Marcus. Voice communication between humans and machines. chapter New Trends in Natural Language Processing: Statistical Natural Language Processing, pages 482–504. National Academy Press, Washington, DC, USA, 1994.

[22] Lluís Màrquez, Xavier Carreras, Kenneth C. Litkowski, and Suzanne Stevenson. Semantic role labeling: An introduction to the special issue. *Comput. Linguist.*, 34(2):145–159, jun 2008.

[23] Fiona Mcneill, Harry Halpin, Ewan Klein, and Alan Bundy. Merging stories with shallow semantics. 04 2006.

[24] Joakim Nivre, Johan Hall, Jens Nilsson, Gülşen Eryiğit, and Svetoslav Marinov. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of the Tenth Conference on Computational Natural*

Language Learning, CoNLL-X '06, pages 221–225, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[25] Kemal Oflazer. Dependency parsing with an extended finite-state approach. *Comput. Linguist.*, 29(4):515–544, December 2003.

[26] Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür. *Building a Turkish Treebank*, pages 261–277. Springer Netherlands, Dordrecht, 2003.

[27] Seong-Bae Park and Byoung-Tak Zhang. Text chunking by combining hand-crafted rules and memory-based learning. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 497–504, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[28] Muhammet Şahin, Umut Sulubacak, and Gülşen Eryiğit. Redefinition of turkish morphology using flag diacritics. In *Proceedings of The Tenth Symposium on Natural Language Processing (SNLP-2013)*, Phuket, Thailand, October 2013.

[29] Khaled Shaalan. Rule-based approach in Arabic natural language processing. *The International Journal on Information and Communication Technologies (IJICT)*, 3(3):11–19, 2010.

[30] Svetlana Stenchikova, Dilek Hakkani-Tur, and Gokhan Tur. Qasr: question answering using semantic roles for speech interface., 01 2006.

[31] L. Tesnière, T. Osborne, and S. Kahane. *Elements of Structural Syntax*. John Benjamins Publishing Company, 2015.

[32] GuoDong Zhou and Jian Su. Named entity recognition using an HMM-based Chunk Tagger. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 473–480, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.