EXTRACTION OF GRAMMAR RULES AND RECURRING PATTERNS IN TURKISH TEXTS BY USING UNSUPERVISED LEARNING ALGORITHMS

by

Nazife ÇEVİK

July 2009

EXTRACTION OF GRAMMAR RULES AND RECURRING PATTERNS IN TURKISH TEXTS BY USING UNSUPERVISED LEARNING ALGORITHMS

by

Nazife ÇEVİK

A thesis submitted to

The Graduate Institute of Sciences and Engineering

of

Fatih University

in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Engineering

July 2009

Istanbul, Turkey

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Assist. Prof. Tuğrul YANIK

Head of Department

This is to certify that I have read this thesis and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Assist. Prof. Zeynep ORHAN

Supervisor

Examining Committee Members

Assist. Prof. Zeynep Orhan _____

Assist. Prof. Tuğrul Yanık _____

Assist. Prof. Özgür Özdemir _____

It is approved that this thesis has been written in compliance with the formatting rules laid down by the Graduate Institute of Sciences and Engineering.

_____

Assoc. Prof. Nurullah ARSLAN

Director

EXTRACTION OF GRAMMAR RULES AND RECURRING PATTERNS IN TURKISH TEXTS BY USING UNSUPERVISED LEARNING ALGORITHMS

Nazife ÇEVİK

M. S. Thesis - Computer Engineering

July 2009

Supervisor: Assist. Prof. Zeynep ORHAN

ABSTRACT

This study proposes the idea on building an automatic grammar extraction in Turkish Texts by using unsupervised learning algorithms. There are three types of text citation chosen to obtain a corpus in this study: *stories, magazines and newspaper articles*, which are found to be the most frequently encountered content.

This corpus is assumed to contain Turkish sentences that are grammatically correct. By using unsupervised learning algorithms, the grammar structure of these sentences is taught to the computer. Additionally, by using grouping method, Context Free Grammar and Parse Tree of Turkish language is generated.

Although testing results demonstrate that there are several problems with grammar extraction, high accuracy is achieved. This study should be an initial process for further applications dedicated for Turkish language.

Keywords: Grammar Extraction, Context Free Grammars, Parse Tree, Natural Language Processing

TÜRKÇE METİNLERDEKİ GRAMER KURALLARININ VE TEKRARLI KALIPLARIN ÖĞRETİCİSİZ ÖĞRENME ALGORİTMALARI YARDIMIYLA ÇIKARILMASI

Nazife ÇEVİK

Yüksek Lisan Tezi – Bilgisayar Mühendisliği

July 2009

Tez Yöneticisi Yrd. Doç.Dr. Zeynep ORHAN

ÖZ

Bu çalışma, Türkçe metinlerden öğreticisiz öğrenme algoritmalarını kullanarak otomatik gramer çıkarımı projesinin nasıl yapıldığını anlatmaktadır. En sık karşılaşılan içeriğe sahip olduğu görülen hikâye, dergi ve gazete makalelerinden alıntılar yapılarak, Türkçe metinler oluşturulmuştur.

Oluşturulan bu metinlerin doğru gramer yapısına sahip olduğu varsayılmıştır. Öğreticisiz öğrenme algoritmaları kullanılarak, bu metinlerin gramer yapıları bilgisayara öğretilmiştir. Buna ek olarak, gruplama metodu kullanılarak, Türkçe dili için İçerikten Bağımsız Gramer ve Çözümleme Ağacı oluşturulmuştur.

Test sonuçları gramer çıkarımında bazı hatalar olduğunu göstersede, yüksek doğruluk oranına sahip bir başarı elde edilmiştir. Bu çalışma Türkçe dili için ileride geliştirilebilecek projelere yardımcı olacak bir projedir.

Anahtar Kelimeler: Gramer Çıkarımı, İçerikten Bağımsız Gramer, Çözümleme Ağaçları, Doğal Dil İşleme

DEDICATION

To my family

ACKNOWLEDGEMENT

This work would not have been possible without the support and encouragement of my adviser, Assist. Prof. Zeynep ORHAN who has also assisted me in numerous ways. I attribute the level of my Masters degree to her encouragement and effort and without her this thesis would not have been completed or written.

I would also like to thank Ahmet Akın who is the project manager of Zemberek Library which I took many help for my thesis.

Finally, I thank my parents for supporting me throughout all my studies at the university.

TABLE OF CONTENTS

ABSTRACT III
ÖZ IV
DEDICATION V
ACKNOWLEDGEMENT VI
TABLE OF CONTENTS VII
LIST OF TABLES X
LIST OF FIGURES XII
CHAPTER 1 1
INTRODUCTION 1
CHAPTER 2 4
AUTOMATIC GRAMMAR EXTRACTOIN 5
2.1 BRIEF HISTORY OF NATURAL LANGUAGE PROCESSING 5
2.2 LEVELS OF NATURAL LANGUAGE PROCESSINGS 7
2.2.1 Phonology 7
2.2.2 Morphology 8
2.2.3 Lexical 8
2.2.4 Syntactic 9
2.2.5 Semantic 9
2.3 MORPHOLOGICAL ANALYSIS 10
2.3.1 Morpheme 10
2.3.2 Derivational Morphemes vs Inflectional Morphemes 12
2.4 CONTEXT FREE GRAMMARS 12
2.4.1 Example for a CFG 15

2.4.2 A Limited Subset of English Grammar 16
2.4.3 Some Grammar Rules for Turkish 18
2.4.3.1 Some Alphabetical List of Turkish Suffixes 19
2.4.3.3 Little Grammar for Turkish 22
2.5 PARSING 24
2.6 EVALUATION MEASURES 25
2.7 AMBIGUITY 26
2.7.1 Ambiguous CFGs 28
2.7.2 Turkish Morphological Disambiguation 29
CHAPTER 3 31
AUTOMATIC TURKISH GRAMMAR EXTRACTION 31
3.1 METHODOLOGY 31
3.2 CORPUS-BASED APPROACH 32
3.3 MORPHOLOGICAL ANALYSIS OF SENTENCES 33
3.3.1 Zemberek Library 33
3.3.2 Morphological Analysis of Turkish Words 34
3.4 NUMBER OF WORD TYPES AND SUFFIXES 36
3.5 ASCENDING ORDER OF WORD TYPES 36
3.6 CFG EXTRACTION OF WORDS 37
3.7 PROPOSED METHOD FOR GRAMMAR EXTRACTION 39
3.8 CFG EXTRACTION OF SENTENCES 43
3.9 SUBSET OF CFG GENERATION OF TURKISH SENTENCES 44
3.10 TEST RESULTS 51
3.11 GENERATE WORD AND SUFFIX 51
3.12 GENERATION OF TURKISH SENTENCES 52
CHAPTER 4 55
CONCLUSION 55
4.1 EVALUATION OF RESULTS 55

4.2 FUTURE WORK.....56
REFERENCES 57

LIST OF TABLES

TABLE

2. 1 Morpheme for English and Turkish 11
2. 2 Morphemes 11
2. 3 Derivational and Inflectional Morphemes 12
2. 4 Formal Definition of CFG 14
2. 5 A Part of English Grammar 17
2. 6 CFG for Sentences 17
2. 7 Suffixes and Its Explanation 19
2. 8 Extra Letters 21
2. 9 CFG to Parse Tree 25
2. 10 Confusion Matrix 26
2. 11 Formula for Confusion Matrix 26
2. 12 Word and Its Parses 30
3. 1 Number of Sentences, Length and Citation 32
3. 2 Word Type / Frequency of the Corpus 32
3. 3 Morphologic Analysis of a Sentence 34
3. 4 Morphological Analysis of a Sentence with Ambiguity 35
3. 5 Number of Types of Words 36
3. 6 Ascending Order of Word Types / Word + Suffixes 37
3. 7 Extraction of Words 38
3. 8 Grouped Rules 42
3. 9 Number of Grouped Rules and Probability 43

3. 10 Turkish Sentence Extraction 44
3. 11 CFG of Turkish Language 45
3. 12 Accuracy Rate 51
3. 13 Production of Turkish Words 52
3. 14 Generated Sentences 53

LIST OF FIGURES

FIGURE

2. 1 Parsing of Palindrome 15
2. 2 Parse Tree of the NP 16
2. 3 Parse Tree of an English Sentence 18
2. 4 Parse Tree of the Rule SıfatIsım 23
2. 5 Parse Tree of a Turkish Sentence 23
2.6 Parsing Example 24
2. 7 Example of Parse Tree 25
2. 8 First Structural Ambiguity 28
2. 9 Second Structural Ambiguity 28
2. 10 Two Parse Trees of Word *ab* Using a Grammar 29
3. 1 Zemberek Outlook 33
3. 2 Grouping 40
3. 3 An Example for Grouping 41

CHAPTER 1

INTRODUCTION

Natural language processing (NLP) is a field of computer science concerned with the interactions between computers and human (natural) languages. Natural language generation systems convert information from computer databases into readable human language. Natural language understanding systems convert samples of human language into more formal representations that are easier for computer programs to manipulate. Many problems within NLP apply to both generation and understanding; for example, a computer must be able to model morphology (the structure of words) in order to understand an English sentence, but a model of morphology is also needed for producing a grammatically correct English sentence[1].

NLP has significant overlap with the field of computational linguistics, and is often considered a sub-field of artificial intelligence. The term natural language is used to distinguish human languages (such as Turkish, Spanish, Swahili or Swedish) from formal or computer languages (such as C++, Java or LISP). Although NLP may encompass both text and speech, work on speech processing has evolved into a separate field.

'Naturally occurring texts' can be of any language, mode, genre, etc. The texts can be oral or written. The only requirement is that they be in a language used by humans to communicate to one another. Also, the text being analyzed should not be specifically

[1] http://en.wikipedia.org/wiki/Natural_language_processing

constructed for the purpose of the analysis, but rather that the text is gathered from actual usage (Liddy, 2003)

The notion of ‘levels of linguistic analyses refer to the fact that there are multiple types of language processing known to be at work when humans produce or comprehend language. It is thought that humans normally utilize all of these levels since each level conveys different types of meaning. But various NLP systems utilize different levels, or combinations of levels of linguistic analysis, and this is seen in the differences amongst various NLP applications. This also leads to much confusion on the part of non-specialists as to what NLP really is, because a system that uses any subset of these levels of analysis can be said to be an NLP-based system. The difference between them, therefore, may actually be whether the system uses ‘weak’ NLP or ‘strong’ NLP (Liddy, 2003).

Natural language processing provides both theory and implementations for a range of applications. In fact, any application that utilizes text is a candidate for NLP. The most frequent applications utilizing NLP include the following:

Information Retrieval – given the significant presence of text in this application, it is surprising that so few implementations utilize NLP. Recently, statistical approaches for accomplishing NLP have seen more utilization, but few systems other than those have developed significant systems based on NLP.

Information Extraction (IE) – a more recent application area, IE focuses on the recognition, tagging, and extraction into a structured representation, certain key elements of information, e.g. persons, companies, locations, organizations, from large collections of text. These extractions can then be utilized for a range of applications including question-answering, visualization, and data mining.

Supervised learning involves discrimination, meaning it uses weights and labeled data used as comparisons. Training begins by supplying the system with input patterns and their actual outputs. The system performs calculations and generates a predicted output. When these comparisons are done the network remembers the differences, or the errors, between the actual and predicted outputs. The weights are then adjusted to minimize the

difference between the two outputs. This procedure is repeated until one of two things happens (Powell, 2008). Supervised methods clearly outperform unsupervised ones, but they are much more time consuming and in many cases it's impossible to find a treebank or corpus, suitable for a specific task (Volsky and Meroz, 2008).

Unsupervised learning allegedly involves no target values. In fact, for most varieties of unsupervised learning, the targets are the same as the inputs. In other words, unsupervised learning usually performs the same task as an auto-associative network, compressing the information from the inputs.[2]

Unsupervised learning is very popular for NLP problems, including Machine Translation (MT), Text Summarization (TS), Information Extraction (IE), etc. Grammar rules and recurring patterns are needed in many NLP applications. However, obtaining the complete and sound set of these rules and patterns is really a very difficult, time-consuming and error-prone task. The methods that lead to the automatic extraction of the sets that are in consideration more advantageous and gaining an increased attention in the NLP researches. In the internet era, not only the enormous amounts of raw texts, but annotated and pre-processed texts are available in numerous electronic formats. By using these texts, general patterns and structures that form the components of grammar rules can be produced by using statistical methods. In order to test the accuracy and correctness of automatically extracted structures, new sentences can be generated out of them. The outcome obtained in this study this way can easily be used in and shed light to various NLP applications.

Turkish is a language that has been widely used and has an important role among the world languages. Today, it has been spoken with different accents and dialects in more than 20 different geographical areas over the world. Turkic languages are spoken by some 180 million people as a native language; and the total number of Turkic speakers is about 200 million, including speakers as a second language. Despite of the interdisciplinary applications, such as computational linguistics (CL), natural language processing (NLP), artificial intelligence, etc. that have gained increasing attention in the world and its

[2] http://www.faqs.org/faqs/ai-faq/neural-nets/part2/section-22.html

common usage, Turkish is a lesser studied language in these fields. Although there are studies about grammar rules and extractions for most of the languages such as English, Russian and Japanese, there is no application for Turkish language unfortunately. From an engineering perspective, the extracted grammars are valuable resources for NLP applications, such as parsing, computational lexicon development, and machine translation (MT), to name a few.

CHAPTER 2

AUTOMATIC GRAMMAR EXTRACTOIN

2.1 BRIEF HISTORY OF NATURAL LANGUAGE PROCESSING

Avram Noam Chomsky (1928-) and his followers have transformed linguistics. Indeed, despite many difficulties and large claims later retracted, the school of deep or generative grammar still holds centre stage. Chomsky came to prominence in a 1972 criticism of the behavourist's B.F. Skinner's book *Verbal Behaviour*. Linguistic output was not simply related to input. Far from it, and a science which ignored what the brain did to create its novel outputs was no science at all. Chomsky was concerned to explain two striking features of language — the speed with which children acquire a language, and its astonishing fecundity, our ability to create a endless supply of grammatically correct sentences without apparently knowing the rules. How was that possible? Only by having a) an underlying syntax and b) rules to convert syntax to what we speak. The syntax was universal and simple. A great diversity of sentences can be constructed with six symbols. Take *cats sits on the mat.* Older readers will remember their parsing exercises at school: indefinite article, noun, verb, preposition, definite article, and noun. Chomsky uses a similar approach but his "parsing" applies to all languages. But how we convert to *the mat was sat on by a cat*? The answer, argued Chomsky, were innate transformation rules by which a fundamental deep structure is converted to the surface sentence. Matters are not usually so straightforward, of course, and the rules can be very complex indeed, but Chomsky and his coworkers have now provided them.

If many languages are now classified along Chomsky lines, why hasn't the approach entirely swept the board, bringing all linguists into the fold of orthodoxy? First there are procedural problems. The American behaviourists, and more so the London school, had a very thorough training in gathering field evidence. Speech was what native speakers actually spoke, not what the anthropologist thought they might accept as correct usage. The Chomskians use introspection (i.e. the linguists themselves decide whether a sentence is good grammar), an approach which can allow "facts" to be fitted to theory and which has somewhat restricted application to the European languages that Chomskians regard themselves as familiar with. Then there is the matter of laboratory testing. Surface sentences that are generated by the more convoluted transformation rules should take speakers longer to produce. The evidence is somewhat contradictory.

But more important than these are the theoretical issues. What are these deep structures and transformation rules — i.e. are they something "hardwired" into the brain or simply a propensity to perform in ways we can view along Chomskian lines? Chomsky is undecided. And, if the structures are real, is this the philosopher's goal: we can base semantics on deep grammar? Some have done so, though Chomsky himself has now abandoned these hopes. Chomsky is not a Structuralist, and there is more to understanding than the ability to recast sentences — an appreciation of the world outside, for example, which we perceive and judge on past experience.

One interesting development from the London School was that of Sydney Lamb and Peter Reich. Lamb charted language as networks of relationships. By using a very simple set of "nodes" he was able to represent phonology, syntax and semantics, and to explain linguistic patterning at various levels. Reich used computer modelling to simulate this approach and explain the difficulties we experience with multiply embedded sentences — *I spoke to the girl whose mother's cat which I didn't know was run over when she wasn't looking* sort of thing. But neither approach coped properly with the prevailing Chomskian structural picture, and wasn't pursued[3].

[3] http://www.bartleby.com/65/li/linguist.html

2.2 LEVELS OF NATURAL LANGUAGE PROCESSINGS

The most explanatory method for presenting what actually happens within a Natural Language Processing system is by means of the 'levels of language' approach. This is also referred to as the synchronic model of language and is distinguished from the earlier sequential model, which hypothesizes that the levels of human language processing follow one another in a strictly sequential manner. Psycholinguistic research suggests that language processing is much more dynamic, as the levels can interact in a variety of orders. Introspection reveals that we frequently use information we gain from what is typically thought of as a higher level of processing to assist in a lower level of analysis.

For example, the pragmatic knowledge that the document you are reading is about biology will be used when a particular word that has several possible senses (or meanings) is encountered, and the word will be interpreted as having the biology sense. Of necessity, the following description of levels will be presented sequentially. The key point here is that meaning is conveyed by each and every level of language and that since humans have been shown to use all levels of language to gain understanding, the more capable an NLP system is, the more levels of language it will utilize (Liddy, 2001).

2.2.1 Phonology

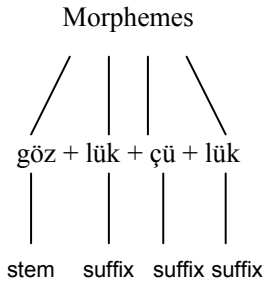
This level deals with the interpretation of speech sounds within and across words. There are, in fact, three types of rules used in phonological analysis: 1) phonetic rules – for sounds within words; 2) phonemic rules – for variations of pronunciation when words are spoken together, and; 3) prosodic rules – for fluctuation in stress and intonation across a sentence. Here are some of the phonemic symbols representing sounds in English[4]:

/ æ / = the 'a' in hat
/ k / = the 'c' in cap

[4] http://www.buzzin.net/english/phonol.htm

2.2.2 Morphology

This level deals with the componential nature of words, which are composed of morphemes – the smallest units of meaning. For example, the word preregistration can be morphologically analyzed into three separate morphemes: the prefix pre, the root registrar, and the suffixation. Since the meaning of each morpheme remains the same across words, humans can break down an unknown word into its constituent morphemes in order to understand its meaning. That is, morphology is the study of the structure and formation of the words. For example:



This example shows that the Turkish word “gözlükçülük” is morphologically analyzed and the stem and the suffixes of the word is found. Not only stem or affixes of the words but also types of the words should be analyzed in morphology.

2.2.3 Lexical

At this level, humans, as well as NLP systems, interpret the meaning of individual words. Several types of processing contribute to word-level understanding – the first of these being assignment of a single part-of-speech tag to each word. In this processing, words that can function as more than one part-of-speech are assigned the most probable part - of speech tag based on the context in which they occur. In Lexical Anaysis, the input is taken as character and converted as tokens (Ullman, 2009).

For example:

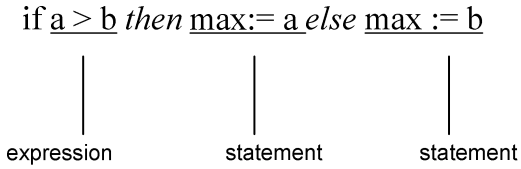
if foo = bar then...

consists of tokens:

IF, <ID,1>, EQSIGN, <ID,2>, THEN...

2.2.4 Syntactic

This level focuses on analyzing the words in a sentence so as to uncover the grammatical structure of the sentence. This requires both a grammar and a parser. The output of this level of processing is a (possibly delinearized) representation of the sentence that reveals the structural dependency relationships between the words. For example, consider a code like:



Here, if the expression is correct, then the statement is executed. So, the purpose of syntactic analysis is to determine the structure of the input text[5].

2.2.5 Semantic

This is the level at which most people think meaning is determined, however, as we can see in the above defining of the levels, it is all the levels that contribute to meaning. Semantic processing determines the possible meanings of a sentence by focusing on the interactions among word-level meanings in the sentence. This level of processing can include the semantic disambiguation of words with multiple senses; in an analogous way to

[5] http://eli-project.sourceforge.net/elionline/syntax_toc.html

how syntactic disambiguation of words that can function as multiple parts-of-speech is accomplished at the syntactic level. Semantic disambiguation permits one and only one sense of polysemous words to be selected and included in the semantic representation of the sentence. For example, in C programming language the break statements can only occur inside switch or loop statements.

2.3 MORPHOLOGICAL ANALYSIS

Morphology is the identification, analysis and description of the structure of words. While words are generally accepted as being (with clitics) the smallest units of syntax, it is clear that in most (if not all) languages, words can be related to other words by rules. For example, English speakers recognize that the words *dog*, *dogs*, and *dog catcher* are closely related. English speakers recognize these relations from their tacit knowledge of the rules of word formation in English. They infer intuitively that *dog* is to *dogs* as *cat* is to *cats*; similarly, *dog* is to *dog catcher* as *dish* is to *dishwasher*. As an example of Turkish; if the words *köpek, köpekler, köpek bakıcısı* are considered, they infer that *köpek* is *to köpekler, kedi* is to *kediler* and *köpek* is to *köpek bakıcısı*. The rules understood by the speaker reflect specific patterns (or regularities) in the way words are formed from smaller units and how those smaller units interact in speech (Lightbrown and Spada, 1993). In this way, morphology is the branch of linguistics that studies patterns of word formation within and across languages, and attempts to formulate rules that model the knowledge of the speakers of those languages[6].

2.3.1 Morpheme

Morpheme is the smallest unit of linguistic meaning. A single word may be composed of one or more morphemes.

[6] http://en.wikipedia.org/wiki/Context Free Grammars

Example:

göz + lük + çü + lük (Turkish)

- Every word in every language is composed of one or more morphemes.

Table 2. 1 Morpheme for English and Turkish

Number of Morpheme	**Turkish**
One Morpheme	kız
Two Morpheme	bil+gi
Three Morpheme	bil + gi + li
Four Morpheme	göz + lük + çü + lük
More than four	Uygar+laş+tır+ama+dık+lar+ ımız+dan+mış+sınız

Table 2. 2 Morphemes

Prefixes	**Suffixes**	**Infixes**	**Circumfixes**
un- uncover (English) no word for Turkish	-er singer (English) -lı yaralı (Turkish)	*fikas* "strong" *fumikas* "to be strong" (Bontoc language)	*chokma* "he is good" *ik + chokm + o* "he isn't good" (Chickasaw Language)

2.3.2 Derivational Morphemes vs Inflectional Morphemes

Table 2. 3 Derivational and Inflectional Morphemes

Derivational Morpheme	**Inflectional Morpheme**
1. Derivational morphemes derive a new word by being attached to root morphemes or stems.	1. Inflectional morphemes contain grammatical information such as number (plural), tense, possession and so on.
2. They can be both suffixes and prefixes in English. Examples: *beauti**ful**, **un**happy* They can be only suffix for Turkish Examples : su**lu**, demir**ci**	2. They are only found in suffixes in English and Turkish. Examples : boy**s**, walk**ed** (English) çocuk**lar,** gel**di** (Turkish)
3. Meaning changes Examples: sing+er (deriving a new word with the meaning of a person who sings in English). koşucu (deriving a new word with the meaning of person who runs in a competition in Turkish).	3. No change of Meaning Examples: walk vs. walks (English) çocuk vs. çocuklar (Turkish)

2.4 CONTEXT FREE GRAMMARS

A formal language is context-free if some context-free grammar generates it. These languages are exactly all languages that can be recognized by a non-deterministic pushdown automaton. Context-free grammars play a central role in the description and

design of programming languages and compilers. They are also used for analyzing the syntax of natural languages[7].

In formal language theory, a context-free grammar (CFG) is a grammar in which every production rule is of the form

$V \rightarrow w$

Where V is a single non-terminal symbol and w is a string of terminals and/or non-terminals (possibly empty).

A context-free grammar G = (V,Σ,R, S) is said to be in Chomsky Normal Form (CNF), if and only if every rule in R is of one of the following forms (Kreysloy, 2001).

1. $A \rightarrow a$, for some $A \in V$ and some $a \in \Sigma$

2. $A \rightarrow BC$, for some $A \in V$ and $B,C \in V \setminus \{S\}$

3. $S \rightarrow \in$

Thus, the difference with arbitrary grammars is that the left hand side of a production rule is always a single nonterminal symbol rather than a string of terminal and/or non-terminal symbols. The term "context-free" expresses the fact that non-terminals are rewritten without regard to the context in which they occur. A CFG is defined by four parameters: N, ∑, R,S (Jurafsky and Martin, 2006). Formal definition of CFG's is shown in Table 2. 4.

[7] http://en.wikipedia.org/wiki/Morphology_(linguistics)

Table 2. 4 Formal Definition of CFG

G = {N, ∑, R, S }	
N	A set of non-terminal sysmbols
∑	A set of terminal sysmbols
R	A set of rules or productions, each of the form A → β. Where A is a non-terminal, β is a string of symbols from the infinite set of strings (∑ U N)*
S	A designated start symbol

A language is defined via the concept of derivation. One string derives another one if it can be rewritten as the second one via some series of rule applications (Jurafsky and Martin, 2006). More formally,

if A→β is a production of P and α and γ are any strings in the set (Σ U N)∗, then we say that αAγ directly derives αβγ, or αAγ => αβγ.

A derivation is the sequence of strings over N produced by a sequence of R rule applications, starting from a start symbol S. Example for derivation is (Webber, 2008):

S ⇒ A<u>B</u> ⇒ AB<u>A</u> ⇒ ABA<u>B</u> ⇒ AB<u>A</u>b ⇒ A<u>B</u>ab ⇒ <u>A</u>bab ⇒ A<u>B</u>bab ⇒ AB<u>A</u>bab ⇒ A<u>B</u>abab ⇒ <u>A</u>bababab ⇒ abababb

The language LG generated by a grammar G is defined as the set of strings composed of terminal symbols which can be derived from the designated start symbol S (Jurafsky and Martin, 2006).

LG = {w|w is in Σ∗ and S $\overset{*}{\Rightarrow}$ w}

The problem of mapping from a string of words to its parse tree is called parsing.

2.4.1 Example for a CFG

$L = \{w\ w^r\}$

L is a language that contains all words which read the same forwards and backwards (Jurafsky and Martin, 2006).

To describe L using a context-free grammar, we must identify the following:

Letters of the alphabet – {a, b} – also called terminals.

Start symbol – S

Non-terminals – For this particular problem, we only need 1 nonterminal - S

Production rules:

S -> λ

S -> aSa

S -> bSb

Word = aa

The word *aa* is a palindrome. It should be recognized by our grammar. We would need to use 2 rules, Rule 1: S -> λ and Rule 2: S -> aSa

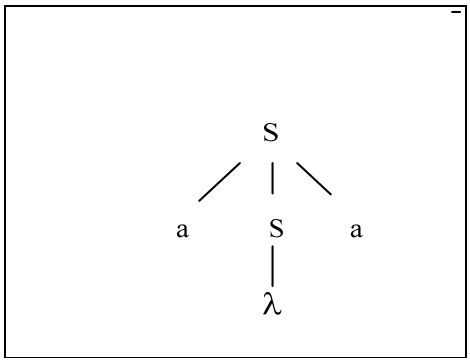


Figure 2. 1 Parsing of Palindrome (Jurafsky and Martin, 2006)

The word "aa" is formed by concatenating all the leaves of the tree together "aλa" which is just "aa".

2.4.2 A Limited Subset of English Grammar

Here is a simple CFG model of noun phrases (NPs) like the trip and a morning flight:

- NP → Det Nom
- Nom → Noun | Noun Nom
- Det → a | the
- Noun → flight | trip | morning

• Individual words like the and flight are called terminal symbols, while categories like NP, Nom, and Noun are called non-terminals. Some examples for English grammar are expressed as following:

Example 1 for using a CFG to generate the NP a flight

- Start with the symbol NP
- Rewrite NP as Det Nom (first rule)
- Rewrite Nom as Noun (second rule)
- Rewrite Det as a (third rule)
- Rewrite Noun as flight (fourth rule)

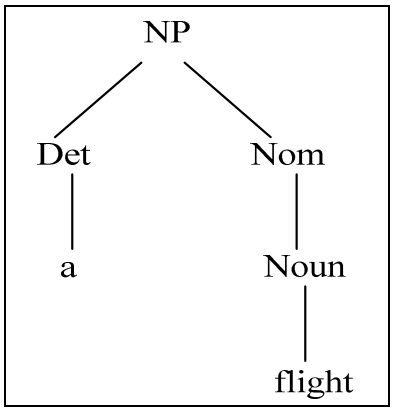


Figure 2. 2 Parse Tree of the NP (Fry, 2004)

Example 2 for a CFG for sentences:

A small part of English Grammar is demonstrated in Table 2. 5 with its extraction respectively. Demonstrated grammar is an example of CFG of English language.

Table 2. 5 A Part of English Grammar

Grammar	Extraction of Grammar
S → *NP VP*	I + want a morning flight
NP → *Pronoun* | *Proper-Noun* | *Det Nominal*	I Los Angeles a + flight
Nominal → *Noun Nominal* | *Noun*	morning + flight flights
VP → *Verb* | *Verb NP* |*Verb NP PP* |*Verb PP*	Do want + a flight leave + Boston + in the morning leaving + on Thursday
PP → *Preposition NP*	from + Los Angeles

Example 3 for a CFG for Sentences (Lexicon Rules):

CFG for English sentences is shown in Table 2. 6. CFG part of the table has completely non-terminals and extraction part of the table has terminals.

Table 2. 6 CFG for Sentences

CFG	Extraction
Noun →	flights | breeze | trip | morning | …
Verb →	is | prefer | like | need | want | fly | …
Adjective →	cheapest | non | stop | f irst | latest | ...
Pronoun →	me | I | you | it | …
Proper-Noun →	Alaska | Baltimore | Los Angeles | …
Determiner →	the | a | an | this | these | that | …
Preposition →	from | to | on | near | …
Conjunction →	and | or | but | …

Example 4 for Generating an English Sentence "I prefer a Morning Flight"

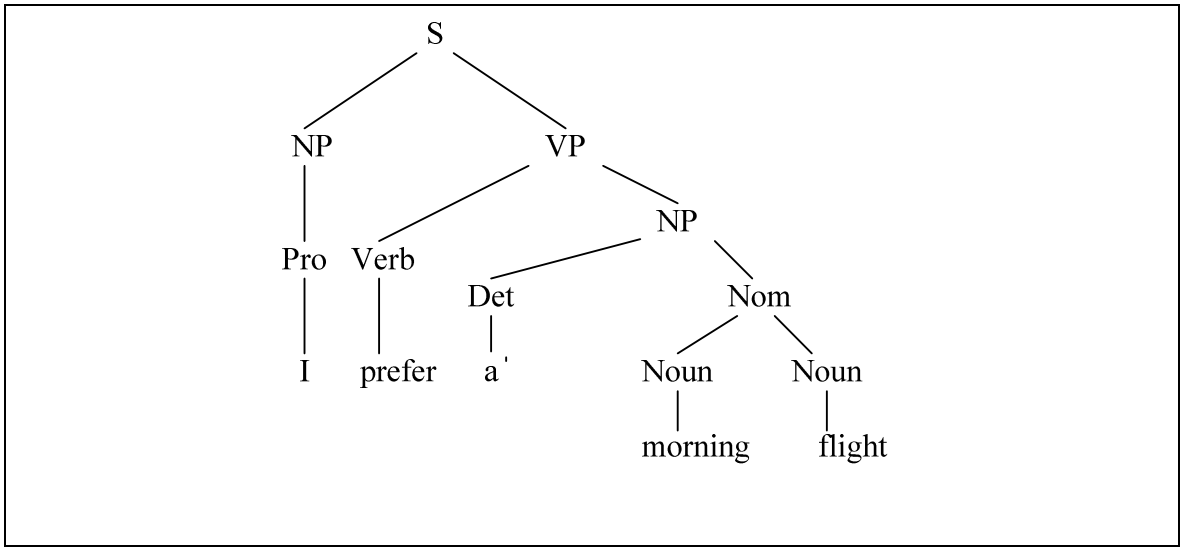


Figure 2. 3 Parse Tree of an English Sentence (Fry, 2004)

2.4.3 Some Grammar Rules for Turkish

Turkish is an agglutinative language, a big word meaning that words have suffixes, possibly several, appended to reflect case and number (for nouns and pronouns) or conjugation (for verbs). So, these pages are mostly collections of tables of suffixes, particularly for the verbs.

This means that this collection of pages might be somewhat mislabeled. Topics like word order and relationships between words or phrases of a sentence, the syntax of the language. But the bulk of the material is about inflection — how words are modified when they play different roles. Inflection includes conjugation of verbs (I *run,* I *am running,* I *was running,* I *ran,* I *had run,* and so on) and inflection of nouns and pronouns (*I, me, my, mine,* and so on, and note that English mostly leaves nouns uninflected).

Other than a few interesting characters, like that undotted "ı", Turkish is fairly straightforward.

2.4.3.1 Some Alphabetical List of Turkish Suffixes

There are some of the suffixes and their explanations in Table 2. 7 for Turkish language.

Table 2. 7 Suffixes and Its Explanation

Suffix	Explanation
-acağım	verb, future general (1st person singular)
-bili-	verb, ability to
-da	noun, locative / prepositional case
-e	verb, gerund verb
-i	noun, direct object / definite case
-lar	noun, plural
-m	noun, owned (by me) in a possession relationship
-r	verb, present positive general (3rd person singular)
-sa-	verb, conditional mood
-t-	verb, causative verb, "to make to" (if stem ends in vowel)
-u	noun, direct object / definite case
-ü	noun, direct object / definite case
-ya	verb, gerund verb

2.4.3.2 Morphology, Character Shifts and Some Irregularities

In most nouns ending in *-ç, -k, -nk* and *-p,* the last consonant lenites (softens or weakens) before a vowel to become *-c, -ğ, -ng* and *-b,* respectively. However, this does not always happen. The word for "tail", "queue", "follower" etc changes as expected to form the accusative *-i/ı/ü/u:*

kuyruk -> kuyruğu

but the word for "law" does not:

hukuk -> hukuku

In some nouns the final consonant geminates (lengthens) before an added vowel. However, this does not always happen. The word for "forgiveness" changes as expected to

form the dative *-e/a:*

af -> affa

but the word for "shelf" does not:

raf -> rafa

Some nouns lost the vowel before a final consonant when a vowel is suffixed. The word for "breast" does when forming the accusative:

koyun -> koynu

but the identically spelled word for "sheep" does not:

koyun -> koyunu

Sometimes that vowel loss is followed by an internal assimilation where the consonant ending of the previous consonant becomes hard or voiceless;

for example, the word for "transcript" or "records":

zabıt -> zaptı

Some nouns end with a back vowel (a/ıo/u) but take a front vowel (e/i/ö/ü) in suffixes. The word for "left" changes as expected forming the dative:

sol -> sola

but the identically spelled word for the musical note does not:

sol -> sole

This may have something to do with the second of those being a rather arbitrary and borrowed "word" or really just a name.

Some pronouns and some compound nouns formed from noun phrases take *n* rather than *y* before a suffixed vowel, and sometimes even before a suffixed consonant. The word for "army" behaves as expected forming the accusative and nominative plural:

ordu -> orduyu

ordu -> ordular

but the word for the plant purslane formed the accustive differently than expected:

semizotu -> semizotunu

and the demonstative pronoun "this" forms both differently:

bu -> bunu

bu -> bunlar

Similarly, the pronoun *o* with its plural *onlar* is the same.

The interrogative pronoun *ne* or "what" is irregular in forming the possessive:

ne -> neyim, but *nem* is also somewhat acceptable

and the genitive:

ne -> neyin (and not the expected **nenin*)

The word for "water", *su*, is similarly irregular. There is a morphologic process of partial reduplication to make Turkish adjectives and adverbs stronger in meaning. Take the beginning of the word through the first vowel, add *m, p, r* or *s*, then repeat the entire adjective. However, there is no pattern for predicting which letter will be used (other than it's always *p* if the adjective starts with a vowel), and sometimes extra letters appear:

Table 2. 8 Extra Letters

yeni	"new"	*yepyeni*	"all new", "very new"
yeşil	"green"	*yemyeşil*	"all green", "very green"
mavi	"blue"	*masmavi*	"all blue", "very blue"
temiz	"clean"	*tertemiz*	"all clean", "very clean"

2.4.3.3 Little Grammar for Turkish

Adjectives (sıfat) are words that describe or modify nouns - A *blue* house, a *rich* man. The adjective in turkish language always comes in front of its noun (isim) as in English.

mavi ev - *the blue house*

mavi evler - *(the) blue houses*

zengin adam - *the rich man*

yorgun çocuklar - *tired children*

Here is a sample CFG model of *SıfatIsim* for Turkish like *güzel kız* :

SıfatIsim → Sıfat + Isim

Sıfat → güzel | iyi | kötü | ...

Isim →Ayşe | Fatma | kız | adam | ...

Individual words like Ayşe and güzel are called terminal symbols, while categories like Sıfat, and Isim are called non-terminals.

Example 1 for Using a CFG to Generate SıfatIsım "güzel kız":

- Start with the symbol *SıfatIsim*
- Rewrite *SıfatIsim* as Sıfat + Isim (first rule)
- Rewrite SIFAT as *güzel* (second rule)
- Rewrite ISIM as *kız* (third rule)

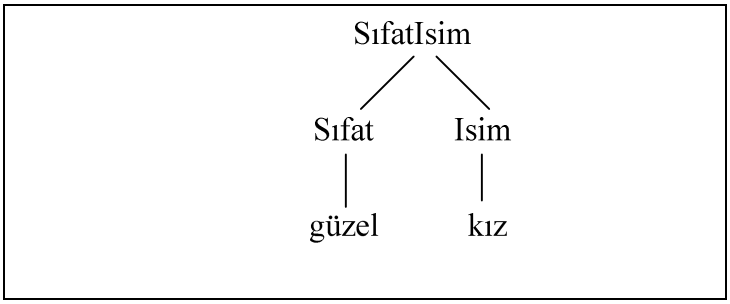


Figure 2. 4 Parse Tree of the Rule SıfatIsım

Example 2 for A CFG for a Sentence "ihtiyar adam geliyor":

SıfatIsimFiil_S_iyor → SIFAT + ISIM + FIIL_S_IYOR

SIFAT → büyük | küçük | güzel | çirkin | ihtiyar | genç | ...

ISIM → kadın | adam | çiçek | petunya | ...

FIIL_S_IYOR → FIIL + FIIL_SIMDIKIZAMAN_IYOR

FIIL → al | gel | gör | git | …

FIIL_SIMDIKIZAMAN_IYOR → -iyor | -ıyor

Example 3 for Generating "ihtiyar adam geliyor":

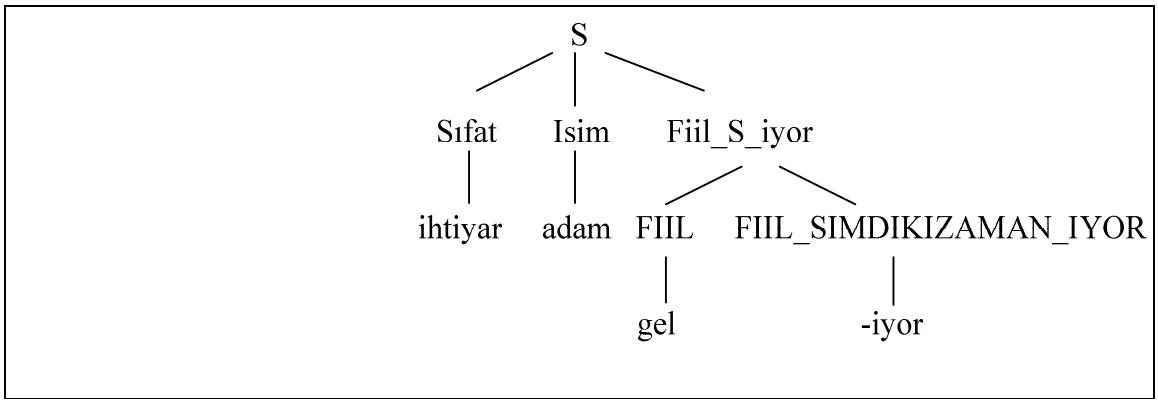


Figure 2. 5 Parse Tree of a Turkish Sentence

2.5 PARSING

A parse tree or concrete syntax tree is an (ordered, rooted) tree that represents the syntactic structure of a string according to some formal grammar. In a parse tree, the interior nodes are labeled by non-terminals of the grammar, while the leaf nodes are labeled by terminals of the grammar. Parse trees may be generated for sentences in natural languages as well as during processing of computer languages, such as programming languages[8].

In a parse tree, each node is either a root node, a branch node, or a leaf node. In the Figure 2. 1 to the following, S is a root node, NP and VP are branch nodes, while Ayşe, took, the, and book are all leaf nodes.

A node can also be referred to as parent node or a child node. A parent node is one that has at least one other node linked by a branch under it. In the example, S is a parent of both NP and VP. A child node is one that has at least one node directly above it to which it is linked by a branch of the tree.

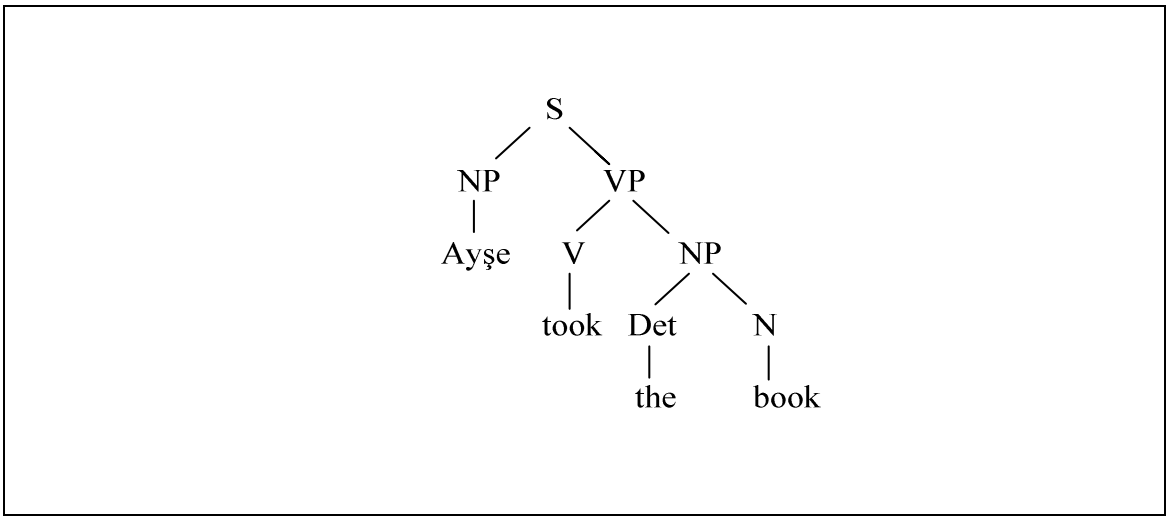


Figure 2.6 Parsing Example

[8] http://en.wikipedia.org/wiki/Parsing

Another example for parse trees shoud be given as a grammar followed by its parse tree as described in following Table 2. 9. Parse tree of the above CFG is shown in Figure 2. 7.

Table 2. 9 CFG to Parse Tree

Rule	Context Free Grammar
An English sentence is composed of a noun phrase, verb and a noun phrase	S → NP V NP
A noun phrase is an article and a noun	NP → A N
A verb is …	V → eat | come | go | …
An article is …	A → A | an | the | …
A noun is …	N → dog | cat | girl | …

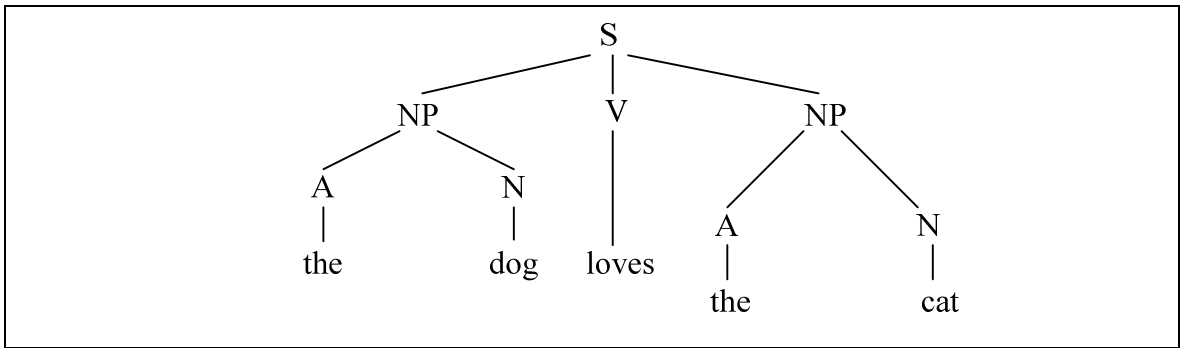


Figure 2. 7 Example of Parse Tree (Webber, 2000)

2.6 EVALUATION MEASURES

The following Table 2. 10 shows a confusion matrix which indicates how predictions on instances are drawn:

Table 2. 10 Confusion Matrix (Lu, 2004)

		Predicted	
		Positive	**Negative**
Known	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

Here, the main idea is that each instance can only be assigned one of two classes: Positive or Negative (e.g. a patient's tumor may be malignant or benign). Each instance (e.g. a patient) has a Known, and a Predicted criteria (Lu, 2004). Following Table 2. 11 can summarize the confusion matrix using various formulas.

Table 2. 11 Formula for Confusion Matrix (Jurafsky and Martin, 2006)

Measure	**Formula**	**Meaning**
Precision	#True Positive / (#True Positive + #False Positive)	#correctly labeled by alg/ all labels assigned by algorithm
Recall	#True Positive / (#True Positive + # False Negative)	#correctly labeled by alg / all possible correct labels
F-Measure	2*precision*recall / (recall+precision)	The F-measure is a balance between the two.

Accuracy	(#True Positive + #True Negative) / N
	N = TP + TN + FP + FN
		Proportion that you got right

2.7 AMBIGUITY

Some grammars, there is no bijective relation between input words and parse tree. While every parse tree corresponds to a single word, the reverse does not hold necessarily. A context free grammar where a word with more than one parse tree exists is called

ambiguous (Pfahler and Fischer, 2008). Ambiguity is rare in English whereas more serious in Turkish. For Turkish 42.1% of the tokens are ambiguous, 1.8% parses per token on average and 3.8% parses for ambiguous tokens (Yüret and Türe, 2008). For example, consider two words of the distinct meanings that exist for the (written) word *tez* in Turkish:

thesis

quick

and the sentences:

Tez yazmaya başladı. (She started to write her thesis)

Tez zamanda geldi. (She came quickly)

From the perspective of human, it is obvious that the first sentence is using the word *tez*, as the meaning of thesis and in the second sentence; the word *tez* is being used as the meaning of quickly. This example is the semantic ambiguity. Other than semantic ambiguity there is structural ambiguity additionally. For example the sentence *John saw the boy who had a telescope* is structurally ambiguos for English. It assigns two different structures as shown in Figure 2. 8 and Figure 2. 9 according to the grammar given below (Frana, 2008).

S → NP (Aux)VP
NP → (D) (AP) N (PP*)
VP → V (NP) (PP*)
PP → P NP

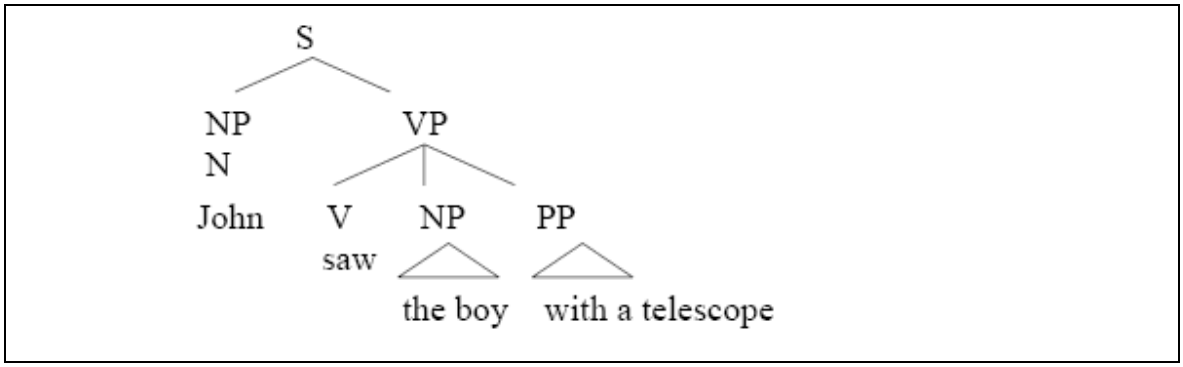


Figure 2. 8 First Structural Ambiguity (Frana, 2008)

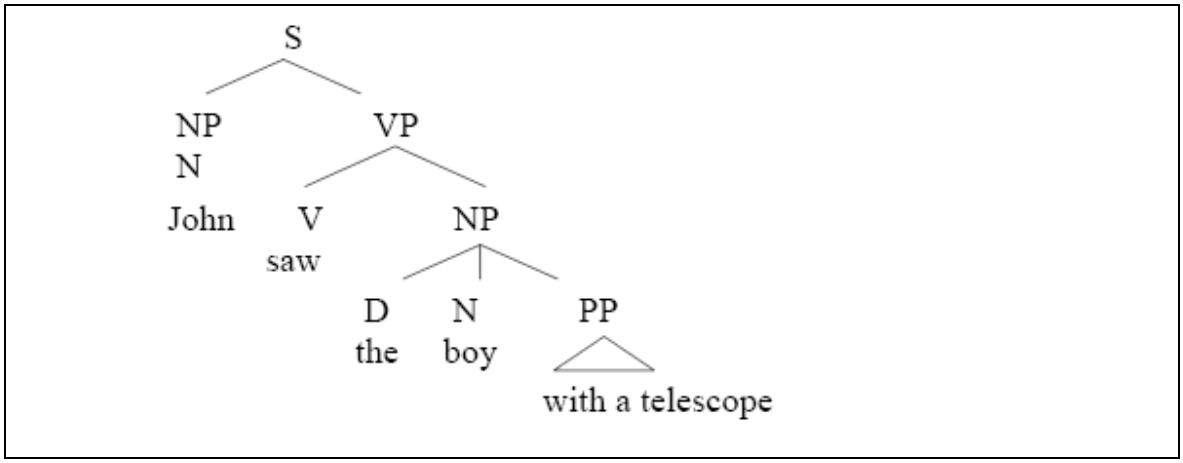


Figure 2. 9 Second Structural Ambiguity (Frana, 2008)

2.7.1 Ambiguous CFGs

An example of a grammar with multiple parse trees is shown as following. It recognizes the word *ab,* which has two different derivations and parse trees.

S → ab
S → Ab
A → a

S → LM ab
S → LM Ab → ab

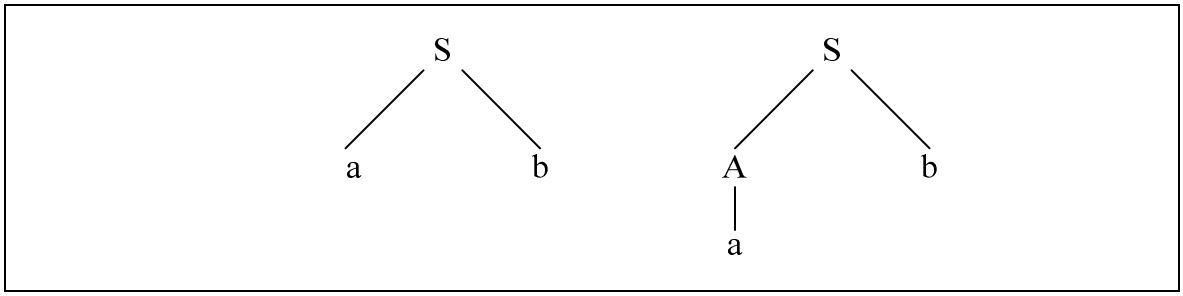


Figure 2. 10 Two Parse Trees of Word *ab* Using a Grammar (Pfahler and Fischer, 2008)

A CFG G = (V, Σ, P, S) is ambiguous, iff a word w ∈ Σ* exists, which has multiple leftmost dervations S ⇒* LM w in G. Using rightmost derivation or parse tree instead results in equivalent definitons (Pfahler and Fischer, 2008).

The ambiguity problem for CFGs is undecidable that is there is no algorithm that solves this problem for every grammar (Pfahler and Fischer, 2008).

2.7.2 Turkish Morphological Disambiguation

The agglutinative or inflective languages such as Turkish impose some difficulties in language processing due to the more complex morphology and relatively free word order in sentences when compared with languages like English. The complex morphology of Turkish allows thousands of word form to be constructed from a single root word using inflectional and derivational suffixes. The example below shows the multiple interpretations for the Turkish word *alın* with their parses as output from a Turkish morphological analyzer (Sak, et al., 2006).

Table 2. 12 Word and Its Parses

Word *alın* and Its Parses
alın+Noun+A3sg+Pnon+Nom **(forehead)**
al+Adj^DB+Noun+Zero+A3sg+P2sg+Nom **(your red)**
al+Adj^DB+Noun+Zero+A3sg+Pnon+Gen **(of red)**
al+Verb+Pos+Imp+A2pl **((you) take)**
al+Verb^DB+Verb+Pass+Pos+Imp+A2sg **((you) be taken)**
alın+Verb+Pos+Imp+A2sg **((you) be offended)**

CHAPTER 3

AUTOMATIC TURKISH GRAMMAR EXTRACTION

This Chapter describes the automatic extraction of Turkish grammar rules. In the first section the architecture of Turkish grammar extraction is explained. Then, several test results of this project are presented.

3.1 METHODOLOGY

In order to generate automatic grammar extraction of Turkish, firstly, a Turkish text that contains several sentences is obtained. All the morphological structure of the sentences is analysed. Then, since unsupervised learning method is proposed for this project, computer learned the correct Turkish grammar rules from the text. In order to achieve this, minimum double, and maximum twenty pairs of words are grouped in a sentence. Then, most frequently used pairs are accepted as the most frequent grammar rule for Turkish. After all, according to those grouped words the Turkish sentences are generated by using automatically generated CFG rules.

3.2 CORPUS-BASED APPROACH

A pre-collected body of Turkish texts called corpus is used rather than studying language by observing language use in actual situation. For this project 1000 sentences in a Turkish corpus is used, which are excerpted from newspaper articles, magazines and stories. The sentences range from 5 words to 15 words in length. The average length is about 13 words. In accordance with this corpus, number of sentences, citation is shown below. The values shown in Table 3. 1 are approximate values.

Table 3. 1 Number of Sentences, Length and Citation

Number of Sentences	**Length**	**Number of Sentences**	**Citation**
206	15	700	Stories
500	10	200	Magazines
100	8	100	Articles
200	5		

Among the sentences of the corpus, frequency of word types have an important role for extraction of grammar rules. Most frequent type of words are shown in Table 3. 2.

Table 3. 2 Word Type / Frequency of the Corpus

Word Type	**Frequency**
Noun	4200
Verb	2270
Adjective	350
Adverb	220

3.3 MORPHOLOGICAL ANALYSIS OF SENTENCES

Morphological analysis of the words of sentences gain importance on account of extracting grammar rules.

3.3.1 Zemberek Library

Zemberek[9] project intends to provide library and applications for solving Turkish Natural Language Processing (NLP) related computational problems. Turkish, by nature has a very different morphological and gramatical structure than Indo-European languages such as English. Since it is an agglutinative language like Finnish even making a simple spell checker is very challenging. By the power of Java, Zemberek tries to overcome this challenge (Zemberek, 2007). An outlook of Zemberek which is taken from the example of morphological analization is shown in Figure 3. 1.

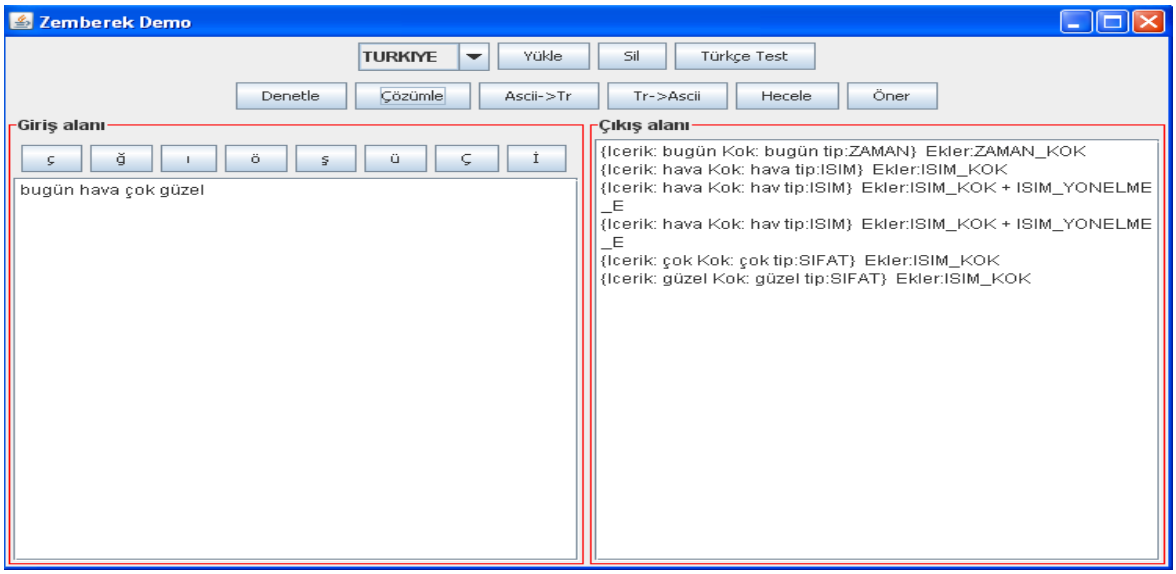


Figure 3. 1 Zemberek Outlook

[9] https://zemberek.dev.java.net

For this project since the words, morphologic structure of the words and some other needings of grammar structure of Turkish language, help of Zemberek library can not be unconsidered.

3.3.2 Morphological Analysis of Turkish Words

Turkish is an agglutinative language with word structures formed by productive affixations of derivational and inflectional suffixes to root words. Morphemes added to a root word or a stem can convert the word from nominal to a verbal structure or vice-versa, or can create adverbial constructs. The surface realizations of morphological structures are restricted and modified by a number of phonetic rules such as vowel harmony (Oflazer, 1992). Table 3. 3 is an example of a sentence word by word which is analysed morphologically from the corpus.

Table 3. 3 Morphologic Analysis of a Sentence

Sentence : kendinden başka kimseyi sevmezdi (she loved nobody except herself)			
Word	**Root**	**Type of Word**	**Suffixes**
kendinden	kendi	ISIM	ISIM_SAHIPLIK_SEN_IN, ISIM_CIKMA_DEN
başka	başka	SIFAT	ISIM_KOK
kimseyi	kimse	ISIM	ISIM_BELIRTME_I
sevmezdi	sev	FIIL	FIIL_OLUMSUZLUK_ME, FIIL_GENISZAMAN_IR, IMEK_HIKAYE_DI
sevmezdi	sev	FIIL	FIIL_DONUSUM_MEZ, IMEK_HIKAYE_DI

Here, the word *sevmezdi* has ambiguity since two different morphologic analysis is seen as in Table 3. 3. So as to solve this ambuguity problem temporary, first occurence of the morphologic analysis is selected as default.

There is another example of a sentence analysed morphologically and has several ambiguous words. As seen in Table 3. 4, the word *ağacı (tree)* and *aynı(same)* are ambiguous words since the suffix that these words has should differ. For instance, the

suffix *–ı* that the word *ağacı* has, should be *ISIM_TAMLAMA_I (NOUN_SUBORDINATIVE), ISIM_SAHIPLIK_O_I (NOUN_POSSESIVE), ISIM_BELIRTME_I (NOUN_INDICATION)* or *ISIM_ILGI_CI (NOUN_RELEVANCY).*

Table 3. 4 Morphological Analysis of a Sentence with Ambiguity

Sentence : büyük çınar ağacı ve pembe petunya aynı ormanda yaşıyordu (great sycamore tree and pink petunia have been living in the same forest)			
Word	**Root**	**Type of Word**	**Suffixes**
çınar	çınar	ISIM	[ISIM_KOK]
ağacı	ağaç	ISIM	[ISIM_KOK, ISIM_TAMLAMA_I]
ağacı	ağaç	ISIM	[ISIM_KOK, ISIM_SAHIPLIK_O_I]
ağacı	ağaç	ISIM	[ISIM_KOK, ISIM_BELIRTME_I]
ağacı	ağa	ISIM	[ISIM_KOK, ISIM_ILGI_CI]
ve	ve	BAĞLAÇ	[BAĞLAÇ_KOK]
pembe	pembe	SIFAT	[ISIM_KOK]
petunya	petunya	ISIM	[ISIM_KOK]
aynı	aynı	ISIM	[ISIM_KOK]
aynı	ayn	ISIM	[ISIM_KOK, ISIM_TAMLAMA_I]
aynı	ayn	ISIM	[ISIM_KOK, ISIM_SAHIPLIK_O_I]
aynı	ayn	ISIM	[ISIM_KOK, ISIM_BELIRTME_I]
ormanda	orman	ISIM	[ISIM_KOK, ISIM_KALMA_DE]
yaşıyordu	yaşa	FIIL	[FIIL_KOK, FIIL_SIMDIKIZAMAN_IYOR, IMEK_HIKAYE_DI]

Since Turkish has 42,1% ambiguous tokens it is too difficult to give a sentence example without ambiguous words or the corpus.

3.4 NUMBER OF WORD TYPES AND SUFFIXES

Number of word types and number of word + suffix have great importance for this approach since number of types of words and word + suffix support the probability of those among the whole words. Obtaining the probability of the types provides mostly used grammar structure of Turkish language. So, the type that has high probability should be considered as the rule that is used in Turkish grammar generation. A sample of this approach for 1000 sentences is given in Table 3. 5.

Table 3. 5 Number of Types of Words

Types of Words / Word + Suffix	**Number / All Words**
ISIM (Noun)	4200 / 9500
SIFAT (Adj.)	350 / 9500
FIIL + FIIL_SIMDIKIZAMAN_IYOR + IMEK_HIKAYE_DI	18 / 9500
ISIM + ISIM_KALMA_DE	75 / 9500

As seen in Table 3. 5, numbers of types that are used in Turkish grammar have diversification. For this project, the types or suffixes that have highest probability are considered for grammar generation whereas lesser used ones are not.

3.5 ASCENDING ORDER OF WORD TYPES

All word types are written in ascending order so as to see the most used types for obtaining probabilities of those as seen in Table 3. 6.

Table 3. 6 Ascending Order of Word Types / Word + Suffixes

Word / Word + Suffix	Number
ISIM	570
FIIL	550
SIFAT	300
ISIM + ISIM_TAMLAMA_I	110
FIIL + FIIL_GENISZAMAN_IR	90
ISIM + ISIM_KALMA_DE	75
ISIM + ISIM_CIKMA_DEN	62
FIIL + FIIL_SUREKLILIK_EREK	56
ISIM + ISIM_TAMLAMA_IN	45
ISIM + ISIM_BULUNMA_LI + IMEK_HIKAYE_DI	27
ISIM + ISIM_SAHIPLIK_SEN_IN + ISIM_BELIRTME_I	20
FIIL + FIIL_SIMDIKIZAMAN_IYOR + IMEK_HIKAYE_DI	18
FIIL + FIIL_OLUMSUZLUK_ME + FIIL_GENISZAMAN_IR + IMEK_HIKAYE_DI	10
FIIL + FIIL_EDILGENSESLI_N + FIIL_YETERSIZLIK_E + FIIL_OLUMSUZLUK_ME + FIIL_KISI_BEN	9

Noun and verb are the most used types of Turkish language as seen in Table 3. 6. In fact, in the Turkish language almost every sentence has a noun and a verb. Ascending order of the word types demonstrates most frequently used structures of the Turkish language so that analyzing the grammar rules becomes easy. Table 3. 6 shows that ISIM (noun), FIIL (verb), SIFAT (adjective) and ISIM_TAMLAMA_I (noun_subordinative) exist most of the Turkish sentences, so these rules should be considered as rules of Turkish language.

3.6 CFG EXTRACTION OF WORDS

Since Turkish has agglutinative word structures with derivative and infectional, most derivational suffixes take place within a word form. Turkish word structures consist of morphemes concatenated to a root morpheme or to other morphemes (Oflazer, et al., 1994).

Table 3. 7 Extraction of Words

Rule	Extraction
FIIL_DONUSUM_MEZ	FIIL + FIIL_DONUSUM_MEZ
FIIL_EDILGENSESLI_N	FIIL + FIIL_EDILGENSESLI_N
FIIL_GENISZAMAN_IR	FIIL + FIIL_GENISZAMAN_IR
FIIL_ISTEK_E	FIIL + FIIL_ISTEK_E
FIIL_KISI_BEN	FIIL + FIIL_EDILGENSESLI_N + FIIL_YETERSIZLIK_E + FIIL_OLUMSUZLUK_ME + FIIL_KISI_BEN
FIIL_SIMDIKIZAMAN_IYOR	FIIL + FIIL_SIMDIKIZAMAN_IYOR
FIIL_SUREKLILIK_EREK	FIIL + FIIL_SUREKLILIK_EREK
FIIL_YETERSIZLIK_E	FIIL + FIIL_YETERSIZLIK_E
IMEK_HIKAYE_DI	FIIL + FIIL_SIMDIKIZAMAN_IYOR + IMEK_HIKAYE_DI
IMEK_SART_SE	FIIL_GENISZAMAN_IR + IMEK_SART_SE
ISIM_BELIRTME_I	ISIM + ISIM_BELIRTME_I
ISIM_BULUNMA_LI	ISIM + ISIM_BULUNMA_LI
ISIM_CIKMA_DEN	ISIM + ISIM_CIKMA_DEN
ISIM_COGUL_LER	ISIM + ISIM_COGUL_LER
ISIM_ILGI_CI	ISIM + ISIM_ILGI_CI
ISIM_KALMA_DE	ISIM + ISIM_KALMA_DE
ISIM_KISI_BEN_IM	FIIL + FIIL_YETERSIZLIK_E + FIIL_OLUMSUZLUK_ME + FIIL_GENISZAMAN_IR + IMEK_SART_SE + ISIM_KISI_BEN_IM
ISIM_TAMLAMA_IN	ISIM + ISIM_TAMLAMA_IN
ISIM_SAHIPLIK_SEN_IN	ISIM + ISIM_SAHIPLIK_SEN_IN

For example, the word *yaşıyordu* would be extracted as *FIIL + FIIL_SIMDIKIZAMAN_IYOR + IMEK_HIKAYE_DI.* The word starts with a verb root and ends up after two suffixes. Some of the word extractions of Turkish are shown in Table 3. 7. These word extractions are formed as CFG rules at the word level.

3.7 PROPOSED METHOD FOR GRAMMAR EXTRACTION

As every language has its own grammar rules, Turkish has also its peculiar grammar rules. There should be several methods in the cause of extracting grammar rules of a language with several methods.

Probability of the rules are found and less probabilities than a certain treshold are removed up to extracting word groups. The rules that have less probability should be ignored since those kind of rules are not seen very often for the language that is analysed.

Every language has a grammar structure and word order in the cause of making gramatically correct sentences. Turkish has also lots of word orders for making gramatically correct sentences. In stead of analysing all sentence structures of Turkish language, a method to analyse word orders automatically is proposed.

First of all, it is supposed that the sentences that we composed are correct sentences gramatically. The words are segmented and annotated with the suitable rules. Then, each rule are grouped with the rule afterward rule starting from the double grouping to the end of the last rule in a sentence. Figure 3. 2 shows the steps of the groups from starting point to the end as double grouping, triple grouping, etc.

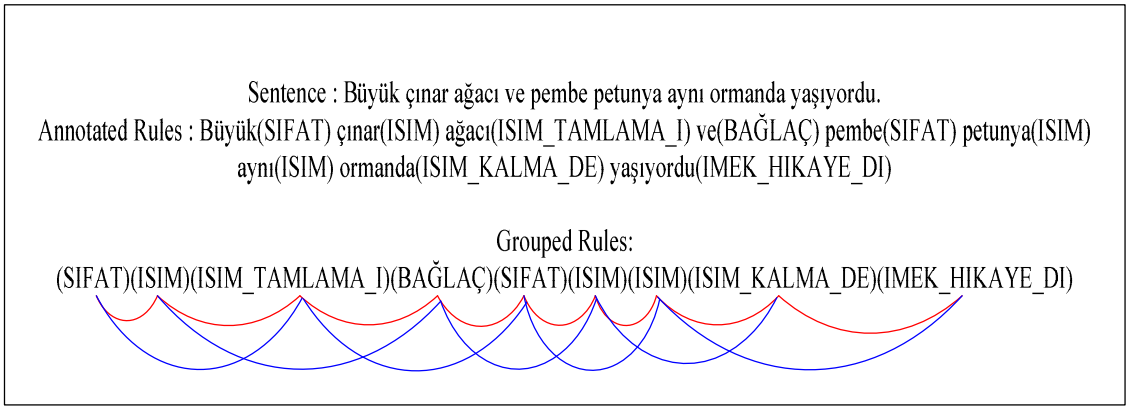


Figure 3. 2 Grouping

Grouping is obtained by the combination of first two morphems as a starting point and ends with last combination words. This method proposes to match every type of word and every rule with the remaining ones. This proves that, a rule can be followed by another rule in a Turkish sentence. As seen in Figure 3. 2, the sentence Büyük çınar ağacı ve pembe petunya aynı ormanda yaşıyordu (Great plane tree and pink petunia have been living in the same forest) gives several possible rule combinations of a Turkish sentence. This example shows us a Turkish sentence should start with an SIFAT (adjective) and should be followed by a ISIM (noun) and etc. So, every possibility of combined rules are analysed and highest ones are taken as possible grammar rules. Figure 3. 3 shows another example of grouped rules for sentences. Table 3. 8 shows several examples of the grouped rules.

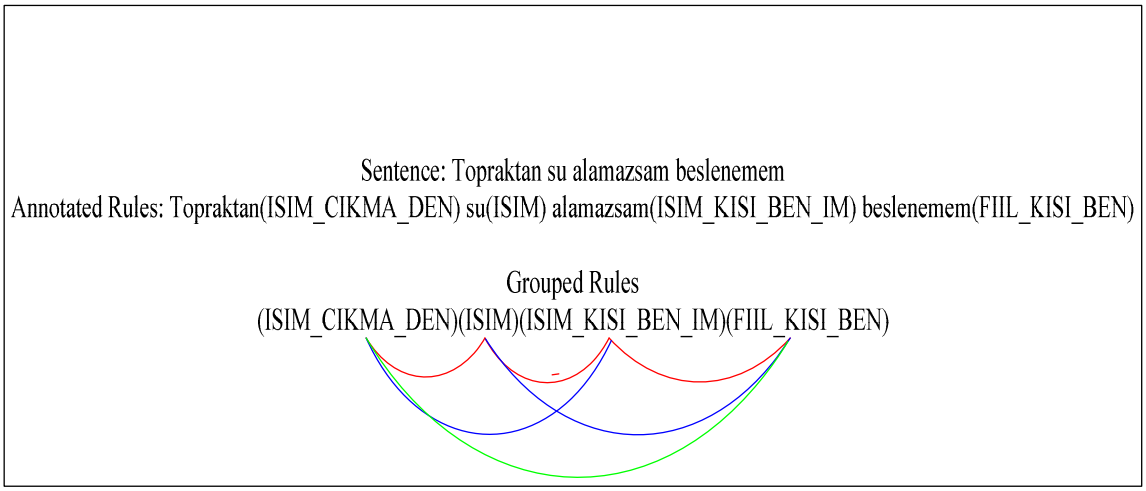


Figure 3. 3 An Example for Grouping

These grouped rules in Figure 3.3 are obtained by the combination of first two morphems as a starting point and ends with last combination words. This method proposes to match every type of word and every rule with the remaining ones. This proves that, a rule can be followed by another rule in a Turkish sentence. As seen in Figure 3. 2, the sentence Büyük çınar ağacı ve pembe petunya aynı ormanda yaşıyordu (Great plane tree and pink petunia have been living in the same forest) gives several possible rule combinations of a Turkish sentence. This example shows us a Turkish sentence should start with an SIFAT (adjective) and should be followed by a ISIM (noun) and etc. So, every possibility of combined rules are analysed and highest ones are taken as possible grammar rules. Figure 3. 3 shows that a Turkish sentence should have these word orders and combinations, that is, in a Turkish sentence for example, rule (SIFAT) should be followed by rule (ISIM) like *büyük ağaç*. Here *büyük* is (SIFAT) and *ağaç* is (ISIM). In the cause of finding most frequently used rules, number of these grouped rules and probability of the rules among all are found as shown in Table 3. 9.

Table 3. 8 Grouped Rules

(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM)
(ISIM_CIKMA_DEN) (ISIM) (ISIM_KISI_BEN_IM) (FIIL_KISI_BEN)
(IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
(ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
(ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
(ISIM) (SIFAT)
(SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
(SIFAT) (ISIM)
(SIFAT) (ISIM) (ISIM)
(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT)
(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT)
(ISIM) (SIFAT) (ISIM_TAMLAMA_I)
(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE) (IMEK_HIKAYE_DI)
(ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
(SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM)
(ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
(IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
(ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)

Table 3. 9 Number of Grouped Rules and Probability

Rule	Number of Groups	Probability
(ISIM) (ISIM)	14	0,116
(ISIM) (SIFAT) (ISIM) (ISIM)	3	0,025
(ISIM) (ZAMIR)	2	0,016
(ISIM) (ISIM_YONELME_E) (SIFAT)	8	0,066
(ISIM) (FIIL_GENISZAMAN_IR)	3	0,025
(ISIM) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)	7	0,058
(ISIM) (SIFAT)	9	0,075
(ISIM) (FIIL)	5	0,041
(ISIM) (ISIM_BELIRTME_I)	5	0,041
(ISIM) (SIFAT) (ISIM)	6	0,05
(SIFAT) (FIIL_GECMISZAMAN_DI)	4	0,033
(ISIM_TAMLAMA_I) (ISIM)	4	0,033
(ZAMAN) (SIFAT)	3	0,025
(ZAMIR) (ZAMAN)	7	0,058
(IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)	14	0,116
(ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)	8	0,066
(ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)	3	0,025
(SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)	9	0,075

3.8 CFG EXTRACTION OF SENTENCES

After word extraction and grouping the rules to have CFG of sentences, CFG extraction of sentence structures is the next and important point of grammar extraction. This part of the project makes clear the grammar rules of Turkish language. Here, names of

the grammar rules are given in accordance with the rules that are extracted from grouped rules. Some of the extraction of sentences are shown in Table 3. 10.

Table 3. 10 Turkish Sentence Extraction

SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ı → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
SıfatIsımIsım_tamlama_ıIsımSıfatIsımIsım → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM)
Isım_cıkma_denIsımIsım_kısı_ben_ımFıil_kısı_ben → (ISIM_CIKMA_DEN) (ISIM) (ISIM_KISI_BEN_IM) (FIIL_KISI_BEN)
Imek_hıkaye_dıIsım_belırtme_ı → (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
Isım_tamlama_ınIsımSıfatIsım_tamlama_ıFıil_sureklılık_erek → (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfat → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
IsımSıfat → (ISIM) (SIFAT)
SıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfat → (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
SıfatIsım → (SIFAT) (ISIM)
SıfatIsımIsım → (SIFAT) (ISIM) (ISIM)
SıfatIsımIsım_tamlama_ıIsımSıfat → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT)
SıfatIsımIsım_tamlama_ıSıfat → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT)
IsımSıfatIsım_tamlama_ı → (ISIM) (SIFAT) (ISIM_TAMLAMA_I)

3.9 SUBSET OF CFG GENERATION OF TURKISH SENTENCES

Whole CFG of Turkish language have not been analysed completely till now. Some rules of word and sentence structure of Turkish language have been analysed and there is not a system that generates grammar automatically from a Turkish text. After all processes such as morphological analysis, CFG extraction of words and grouping techniques for this

project, CFG generation of Turkish language is accomplished by the help of used method. Here some of the CFG rules of Turkish language are shown in Table 3. 11.

Table 3. 11 CFG of Turkish Language

ISIM → petunya | orman | çınar | kibir | ağaç | toprak | su | kimse | orman ...
SIFAT → çok | büyük | pembe | ihtiyar ...
FIIL → ol | yaşa | al | gör | düşün | besle ...
BAĞLAÇ → ve | ama | fakat ...
EDAT → ile | gibi | göre ...
FIIL_DONUSUM_MEZ → FIIL + FIIL_DONUSUM_MEZ
FIIL_EDILGENSESLI_N → FIIL + FIIL_EDILGENSESLI_N
FIIL_GENISZAMAN_IR → FIIL + FIIL_GENISZAMAN_IR
FIIL_ISTEK_E → FIIL + FIIL_ISTEK_E
FIIL_KISI_BEN → FIIL + FIIL_EDILGENSESLI_N + FIIL_YETERSIZLIK_E + FIIL_OLUMSUZLUK_ME + FIIL_KISI_BEN
FIIL_OLUMSUZLUK_ME → FIIL + FIIL_OLUMSUZLUK_ME
FIIL_SUREKLILIK_EREK → FIIL + FIIL_SUREKLILIK_EREK
FIIL_YETERSIZLIK_E → FIIL + FIIL_YETERSIZLIK_E
IMEK_HIKAYE_DI → FIIL + FIIL_SIMDIKIZAMAN_IYOR + IMEK_HIKAYE_DI
ISIM_BELIRTME_I → ISIM + ISIM_BELIRTME_I
ISIM_BULUNMA_LI → ISIM + ISIM_BULUNMA_LI
ISIM_CIKMA_DEN → ISIM + ISIM_CIKMA_DEN
ISIM_COGUL_LER → ISIM + ISIM_COGUL_LER
ISIM_ILGI_CI → ISIM + ISIM_ILGI_CI
ISIM_KALMA_DE → ISIM + ISIM_KALMA_DE
ISIM_KISI_BEN_IM → FIIL + FIIL_YETERSIZLIK_E + FIIL_OLUMSUZLUK_ME + FIIL_GENISZAMAN_IR + IMEK_SART_SE + ISIM_KISI_BEN_IM
ISIM_SAHIPLIK_ONLAR_LERI → ISIM + ISIM_SAHIPLIK_ONLAR_LERI
ISIM_SAHIPLIK_O_I → ISIM + ISIM_SAHIPLIK_O_I
ISIM_SAHIPLIK_SEN_IN → ISIM + ISIM_SAHIPLIK_SEN_IN
ISIM_TAMLAMA_I → ISIM + ISIM_TAMLAMA_I
ISIM_TAMLAMA_IN → ISIM + ISIM_TAMLAMA_IN

SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ı → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
SıfatIsımIsım_tamlama_ıIsımSıfatIsımIsım → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM)
Isım_cıkma_denIsımIsım_kısı_ben_ımFııl_kısı_ben → (ISIM_CIKMA_DEN) (ISIM) (ISIM_KISI_BEN_IM) (FIIL_KISI_BEN)
Imek_hıkaye_dıIsım_belırtme_ı → (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
Isım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erek → (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfat → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
IsımSıfat → (ISIM) (SIFAT)
SıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfat → (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
SıfatIsım → (SIFAT) (ISIM)
SıfatIsımIsım → (SIFAT) (ISIM) (ISIM)
SıfatIsımIsım_tamlama_ıIsımSıfat → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT)
SıfatIsımIsım_tamlama_ıSıfat → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT)
IsımSıfatIsım_tamlama_ı → (ISIM) (SIFAT) (ISIM_TAMLAMA_I)
Isım_tamlama_ınIsım → (ISIM_TAMLAMA_IN) (ISIM)
Isım_tamlama_ıIsımSıfatIsımIsımIsım_kalma_de → (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE)
IsımIsım_tamlama_ıSıfat → (ISIM) (ISIM_TAMLAMA_I) (SIFAT)
Isım_belırtme_ıImek_hıkaye_dı → (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)
IsımIsım_kısı_ben_ım → (ISIM) (ISIM_KISI_BEN_IM)
IsımIsım_tamlama_ıIsımSıfatIsımIsımIsım_kalma_deImek_hıkaye_dı → (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE) (IMEK_HIKAYE_DI)
SıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ırIsım_belırtme_ıImek_hıkaye_dı → (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)
Isım_belırtme_ıIsım_tamlama_ınIsımSıfat →(ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)

IsımSıfatIsım → (ISIM) (SIFAT) (ISIM)
SıfatIsımIsımIsım_kalma_de → (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ın → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsım → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_surekllık_erek → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsım → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM)
SıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ır → (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
Isım_tamlama_ıSıfat → (ISIM_TAMLAMA_I) (SIFAT)
Isım_belırtme_ıIsım_tamlama_ınIsım → (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM)
Isım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ırIsım_belırtme_ı → ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I)
Imek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfat → (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
Isım_kalma_deImek_hıkaye_dı → (ISIM_KALMA_DE) (IMEK_HIKAYE_DI)
IsımIsım_tamlama_ıIsımSıfatIsımIsım → (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_surekllık_erekFııl_genıszaman_ır → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ırIsım_belırtme_ıImek_hıkaye_dı → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)
Isım_tamlama_ıIsımSıfatIsımIsımIsım_kalma_deImek_hıkaye_dı → (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE) (IMEK_HIKAYE_DI)

Isım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ı → (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I)
SıfatIsımIsımIsım_kalma_deImek_hıkaye_dı → (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE) (IMEK_HIKAYE_DI)
Isım_tamlama_ıIsımSıfatIsım → (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM)
IsımSıfatIsımIsım → (ISIM) (SIFAT) (ISIM) (ISIM)
Isım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFıil_sureklılık_erekFıil_genıszaman_ırIsım_belırtme_ı → (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I)
IsımIsım_tamlama_ıIsımSıfatIsım → (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM)
Isım_tamlama_ıIsımSıfatIsımIsım → (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM)
Isım_tamlama_ıIsımSıfat → (ISIM_TAMLAMA_I) (ISIM) (SIFAT)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFıil_sureklılık_erekFıil_genıszaman_ırIsım_belırtme_ıImek_hıkaye_dı → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)
SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ın → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN)
Isım_cıkma_denIsım → (ISIM_CIKMA_DEN) (ISIM)
IsımIsım_cıkma_den → (ISIM) (ISIM_CIKMA_DEN)
IsımIsım_kısı_ben_ımFıil_kısı_ben → (ISIM) (ISIM_KISI_BEN_IM) (FIIL_KISI_BEN)
IsımSıfatIsım_tamlama_ıFıil_sureklılık_erek → (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
Isım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFıil_sureklılık_erek → (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
SıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ı → (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I)
Fıil_genıszaman_ırIsım_belırtme_ıImek_hıkaye_dı → (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)
SıfatImek_hıkaye_dıIsım_belırtme_ı → (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)
Isım_tamlama_ınIsımSıfatIsım_tamlama_ı → (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I)

SıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genısz aman_ır → (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_surek lılık_erekFııl_genıszaman_ırIsım_belırtme_ı → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I)
Isım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ırIsım_belırtme_ıImek_hıkay e_dı → (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)
Isım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ır → (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
Fııl_sureklılık_erekFııl_genıszaman_ır → (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dı → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI)
Imek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erek → (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
Isım_tamlama_ıFııl_sureklılık_erek → (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
Isım_tamlama_ıSıfatImek_hıkaye_dı → (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI)
SıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ın → (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN)
SıfatIsım_tamlama_ıFııl_sureklılık_erek → (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
Isım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık _erekFııl_genıszaman_ır → (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
Isım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık _erek → (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
IsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ırIsım_belırtme_ıImek_hıkaye_dı → (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)

Imek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ın → (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN)
SıfatIsımIsım_tamlama_ıIsım → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM)
SıfatIsımIsım_tamlama_ıIsımSıfatIsımIsımIsım_kalma_de → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE)
SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ır → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR)
SıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erek → (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)
Isım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ırIsım_belırtme_ıImek_hıkaye_dı → (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I) (IMEK_HIKAYE_DI)
Isım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfat → (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ı → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I)
IsımIsım_tamlama_ıSıfatImek_hıkaye_dı → (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI)
Isım_belırtme_ıIsım_tamlama_ın → (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN)
IsımIsım_cıkma_denIsım → (ISIM) (ISIM_CIKMA_DEN) (ISIM)
SıfatIsımIsım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ıIsım_tamlama_ınIsımSıfatIsım_tamlama_ıFııl_sureklılık_erekFııl_genıszaman_ırIsım_belırtme_ı → (SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK) (FIIL_GENISZAMAN_IR) (ISIM_BELIRTME_I)
IsımSıfatIsımIsımIsım_kalma_de → (ISIM) (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE)
Isım_cıkma_denIsımIsım_kısı_ben_ım → (ISIM_CIKMA_DEN) (ISIM) (ISIM_KISI_BEN_IM)
SıfatIsımIsım_tamlama_ı → (SIFAT) (ISIM) (ISIM_TAMLAMA_I)
IsımIsım_tamlama_ı → (ISIM) (ISIM_TAMLAMA_I)

3.10 TEST RESULTS

Experimental results show that automatic grammar extraction of Turkish language is accomplished with maximum %70 accuracy by obtaing morphological analysis, CFGs, parse tree and combination of sequential rules as it is seen in Table 3. 12. This table demonstrates that the number of sentences and rules are same. This means, rules are extracted as the number of sentences. Among these number of sentences and rules, the accuracy rate of those are given according to the whole sentences in the Turkish text.

Table 3. 12 Accuracy Rate

Number of Sentences	***Rule 1***	***Rule 2***	***Rule3***
50	%80	%76	%100
50	%60	%94	%80
50	%100	%90	%84

The results indicate that some rules are hard to be detected automatically. On the other hand, some rules can be completely or at least generally detected without further modifications and this is promising for some other types of generations.

3.11 GENERATE WORD AND SUFFIX

This project proposes the method to generate correct Turkish words so as to generate words of the sentences when the word and suffixes are given. Producing new words by concatenating the word and suffix supports a certain result to test the validity of rules whether generated grammar rules are correct or not. So as to achieve producing Turkish words Zemberek library is used. Zemberek has a functionality for generating Turkish words. When a word and suitable suffix for that word is given with the code, Zemberek produces the word. Generation of words are shown in Table 3.13.

Table 3. 13 Production of Turkish Words

Root	POS	Suffix	Generated Word
ağaç	ISIM	ISIM_TAMLAMA_I	ağacı
gel	FIIL	FIIL_SIMDIKIZAMAN_IYOR + FIIL_KISI_BEN	geliyorum
sev	FIIL	FIIL_GECMISZAMAN_DI + FIIL_KISI_BEN	sevdim
besle	FIIL	FIIL_EDILGENSESLI_N + FIIL_YETERSIZLIK_E + FIIL_OLUMSUZLUK_ME + FIIL_KISI_BEN	beslenemem
kimse	ISIM	ISIM_COGUL_LER + ISIM_BELIRTME_I	kimseleri
orman	ISIM	ISIM_KALMA_DE	ormanda
düşün	FIIL	FIIL_OLUMSUZLUK_ME + FIIL_GENISZAMAN_IR + IMEK_HIKAYE_DI	düşünmezdi

3.12 GENERATION OF TURKISH SENTENCES

Last part of this poject is to generate Turkish sentences automatically by using the generated words. In this part, generated CFGs are used so as to generate Turkish sentences. Rules are analysed for all its terminals and non-terminals. For instance, if analysed rule is *ISIM_TAMLAMA_I*, the rule is fragmented into its terminals or non-terminals. Rule *ISIM_TAMLAMA_I* is fragmented as *ISIM* + *ISIM_TAMLAMA_I* and rule *ISIM* has its terminals as all possible nouns in Turkish language and rule *ISIM_TAMLAMA_I* is the suffix *–i* that comes to the end of a noun which is selected randomly to produce a word with this suffix. Several examples of generation of sentences are shown in Table 3. 14.

Table 3. 14 Generated Sentences

Rule	Extracted Rule	Generated Sentence
SıfatIsimIsım_tamlama_ıSıfat Imek_hıkaye_dıIsım_belırtme_ı	(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)	İhtiyar adam suyu çok görüyordu toprağı
SıfatIsımIsım_tamlama_ı BağlaçSıfatIsımIsım	(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (BAĞLAÇ) (SIFAT) (ISIM) (ISIM)	büyük çınar ağacı ile büyük petunya adam
Isım_cıkma_denBağlaç Isım_kısı_ben_ımFııl_kısı_ben	(ISIM_CIKMA_DEN) (BAĞLAÇ) (ISIM_KISI_BEN_IM) (FIIL_KISI_BEN)	dumandan eğer yaşayamazsam duramam
Imek_hıkaye_dıIsım_belırtme_ı	(IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)	besliyordu kediyi
Isım_tamlama_ınIsımSıfat Isım_tamlama_ıFııl_sureklılık_erek	(ISIM_TAMLAMA_IN) (ISIM) (SIFAT) (ISIM_TAMLAMA_I) (FIIL_SUREKLILIK_EREK)	ayşenin en büyük kedisi sevinerek
IsımIsım_tamlama_ısıfatImek_hıkaye_dı	(ISIM) (ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI)	kadın adamı küçük görüyordu
IsımSıfat	(ISIM) (SIFAT)	Petunya küçük
SıfatImek_hıkaye_dıIsım_belırtme_ı Isım_tamlama_ınIsımSıfat	(SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I) (ISIM_TAMLAMA_IN) (ISIM) (SIFAT)	güzel görüyordu ağacı ormanın petunya büyük
SıfatIsım	(SIFAT) (ISIM)	pembe yaprak
SıfatIsımIsım	(SIFAT) (ISIM) (ISIM)	çok soğuk kimse
SıfatIsımIsım_tamlama_ıIsımSıfat	(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT)	ıslak toprak böceği en çok

SıfatIsımIsım_tamlama_ıSıfat	(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (SIFAT)	ıslak toprak ağacı çok
IsımSıfatIsım_tamlama_ı	(ISIM) (SIFAT) (ISIM_TAMLAMA_I)	orman büyük aynısı
SıfatIsımIsım_tamlama_ıIsımSıfatIsımIsımIsım_kalma_de	(SIFAT) (ISIM) (ISIM_TAMLAMA_I) (ISIM) (SIFAT) (ISIM) (ISIM) (ISIM_KALMA_DE) (IMEK_HIKAYE_DI)	İhtiyar adam ailesi ve çok kimse kendi otelde yaşıyordu
Isım_tamlama_ıSıfatImek_hıkaye_dıIsım_belırtme_ı	(ISIM_TAMLAMA_I) (SIFAT) (IMEK_HIKAYE_DI) (ISIM_BELIRTME_I)	köpeği çok besliyordu sahibi

CHAPTER 4

CONCLUSION

4.1 EVALUATION OF RESULTS

The idea on extracting Turkish grammar rules automatically from Turkish texts is presented in this study.

Provided that the Turkish text has gramatically correct sentences, this study supports correct CFGs of Turkish sentences. Most of the effort for this study is spent for the implementation part. From various stories, magazines and articles a Turkish corpus is obtained. Obtained corpus is analysed morphologically using Zemberek library. Most frequently used types and suffixes are found. At the word level, CFG of words for Turkish are found. Then, at the sentence level by using grouping method, CFG of sentences are found. Results are stored on a file.

While analysing the morphology of the Turkish sentences, some problems are occured originated from Zemberek library. Some of the word types are analysed uncorrectly by Zemberek, as a result the accuracy of generated sentences and some CFG rules have missing parts. Another problem is the frequent ambiguous word structure of Turkish language. Since Turkish has so many ambiguous words, Zemberek analyses all possible types and suffixes of the words. According as, when an implementation of type of word or the suffix is needed, the first occurence is considered, however, it causes incorrect analyses. For example, the word “ağacı” is ambiguous for Turkish since suffix *–i* should be ISIM_TAMLAMA_I, ISIM_SAHIPLIK_O_I, ISIM_BELIRTME_I or ISIM_ILGI_CI.

Here, first occurence which is ISIM_TAMLAMA_I is considered as the correct one. Additionally, although Turkish word “ve” is a conjunction, Zemberek analyses it as noun.

When a word is generated and the original structure would be “ağacın yaprağı (*leaf of tree*)“, however, in stead of noun “ağaç”, conjunction “ve” is used as “venin yaprağı (*and of tree*)”. If these problems are solved, then whole CFG rules and production of the sentences would be completely correct.

4.2 FUTURE WORK

First of all, problem of incorrectly analysed morphological words should be solved. These words should be determined by controlling the output results and should be put a control in order to correct the misssing parts.

Secondly, morphological analyses has Turkish ambiguous words, this ambiguity problem should be solved to determine which word corresponds to which suffix.

Thirdly, performance of this project is not efficient and used corpus has not enough sample sentences, so performance should be improved and the corpus should be enlarged as a future work.

REFERENCES

Collins, M. J., "Three generative, lexicalised models for statistical parsing", *In Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL-97)*, pages 16–23, 1997.

Çakıcı, R., *A Computational Interface for Syntax and Morphemic Lexicons,* M.S. Thesis, Middle East Technical University, 2002.

Frana, I., Structural Ambiguity, 2008, http://people.umass.edu/ilaria/ling201_spring08/teaching_page_files/structural%20ambiguity.pdf

Fry, J., *Context Free Grammar for English,* San Jose State University, California, 2004.

Goldman, N., *Computer Generation of Natural Language from a Deep Conceptual Base,* Ph.D. Thesis, Yale University, 1974.

Hakkani, D. Z., *Design and Implementation of a Tactical Generator for Turkish, a Free Constituent Order Language,* M.S. Thesis, Bilkent University, 1996.

Hearst, M., "Applied Natural Language Processing Lecture notes", Slides adopted by Cohen, W., 15 Nowember 2006, http://www.sims.berkeley.edu/~hearst

Hoffman, B., *The Computational Analysis of the Syntax and Interpretation of Free Word Order in Turkish,* Ph.D. Thesis, University of Pennsylvania, 1995.

Hoffman, B., "A CCG Approach to Free Word Order Languages", *In Proceesings of the 30th Annual Meeting of the Association for Computational Linguistics,* Dublin, Ireland, 1992.

Jurafsky, D., Martin, J.H., *Speech and Language Processing,* New Jersey, 2006.

Korkmaz, T., *Turkish Text Generation with Systemic-Functional Grammar,* M.S. Thesis, Bilkent University, 1996.

Kreysloy, O., "Chomsky Normal Form Lecture Notes", pdf adopted by Kreysloy, O., 23 April 2001, http://www.enseignement.polytechnique.fr/informatique/profs/Luc.Maranget/IF/09/chomsky.pdf

Liddy, E.D., *Natural Language Processing*, Syracuse University, New York, 2003.

Liddy, E.D., *Natural Language Processing,* Encyclopedia of Library and Information Science, New York, 2001

Lightbrown, P., Spada, N., *How Languages are Learned*, Oxford University Press, New York, 1993.

Lu, Z., "Predicting Subcellular Localization of Proteins using Machine-Learned Classifiers, Bioinformatics", Volume 20, pp. 547 – 556, 2004.

McDonald, D.D., "Natural Language Generation", In S.C. Shapiro (Eds), *Encyclopedia of Artificial Intelligence,* pp. 642-655, Wiley, J., 1987.

Oflazer, K., "Two Level Description of Turkish Morphology", *In Proceedings of EACL'93,* Utrecht, Netherlands, 1993.

Oflazer, K., Bozşahin, H.C., "Natural Language Processing Initiative: An Overview", 1993.

Oflazer, K., Çetinoğlu, Ö., Say, B., "Integrating Morphology with Multi-Word Expression Processing in Turkish", pp.130-135, 1994.

Pfahler, P., Fischer, M., *Ambiguity Detections for Context Free Grammars in Eli,* M.S. Thesis, Paderborn University, 2008.

Powell, A. P., *A Comparitive Study of Supervised and Unsupervised Learning Methods in Forecasting The U.S. 30-Year Treasury Bond Yield,* M.S. Thesis, Florida State University, 2008.

Reiter, E., “Building Natural-language Generation Systems”, *In Proceedings of AIPE Workshop*, 1995.

Sak, H., Güngör, T., Saraçlar, M., “Morphological Disambiguation of Turkish Text With Perceptron Algorithm”, Vol. 65, pp. A156-160, 2006.

Ullman, J. D., “Lexical Analysis Lecture Notes”, Slides adopted by Ullman, J.D., 2009, http://infolab.stanford.edu/~ullman/dragon/slides1.pdf

Webber, A., “Grammars and Parse Trees I”, Slides adopted by Webber, A., 2000, http://www.cs.uwm.edu/classes/cs631/slides/02BNF.pdf

Webber, B., “Derivations Lecture Notes”, Slides adopted by Frank. K., 30 September 2008, http://www.inf.ed.ac.uk/teaching/courses/inf2a/slides/2008_inf2a_L04_slides.pdf

Wolsky, E., Meroz, Y., *Unsupervised Learning of Natural Languages*, 2008, http://kybele.psych.cornell.edu/~edelman/TAU-05/Eitan-week11.ppt#285,3,Slide%203

Yüret, D., Türe, F., *Learning Morphological Disambiguation Rules for Turkish*, 2008, http://www.denizyuret.com/pub/hlt-naacl-06/hlt06.ppt#265,%2010,Morphological%20disambiguation