INVESTIGATION OF AUTOMATICALLY DERIVED SUBWORD UNITS FOR TURKISH LVCSR

by

Tuncay Aksungurlu

B.S. in Electronics and Telecommunication Eng., Kocaeli University, 2005

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University

2008

ACKNOWLEDGEMENTS

I would like to thank Assistant Prof. Murat Saraçlar for his valuable support during this difficult process. He always believed in me even when I gave up. I would never have completed this thesis without his encouraging leadership.

I would also thank to Prof. Fikret Gürgen and Assistant Prof. Kerem Harmancı for their participation in my thesis comittee.

I am really grateful of having such a valuable family. They always encouraged me to go on trying in the worst cases.

I am also grateful to all the people at BUSIM for their company.

I would like to thank TUBITAK for their support under the BIDEB 2210 program. I must also emphasize that the database which was used in the experiments, Turkish Broadcast News Database, was built with the support of TUBITAK project number 105E102.

ABSTRACT

INVESTIGATION OF AUTOMATICALLY DERIVED SUBWORD UNITS FOR TURKISH LVCSR

In this thesis, we performed large vocabulary continuous speech recognition (LVCSR) experiments using language models that are built upon different recognition units in order to create a suitable and successful language modeling scheme for Turkish. Since Turkish is an agglutinative language, how you build the language model drastically affects the recognition performance. Whereas traditional word based language models give satisfactory results for English; they do not work well for Turkish due to the inductive morphology. Different language modeling strategies, mainly based on sub-word units like morphemes and stem-endings, are proposed in order to overcome this problem. In this work, the sub-words that are derived in an unsupervised manner, are investigated. Segmentation obtained using different approaches are compared due to their performance in speech recognition. The best WER that has been obtained is 25.24 whereas it has been obtained as 26.90 using the word-based language models.

ÖZET

TÜRKÇE GDSKT İÇİN OTOMATİK OLARAK ELDE EDİLMİŞ KELİME ALTI MODELLERİN İNCELENMESİ

Bu tezde, Türkçe için uygun ve başarılı bir dil modeli yaratma yaklaşımı gerçekleştirebilmek amacıyla, farklı tanıma birimleri temel alınarak oluşturulmuş dil modelleri kullanılarak geniş dağarcıklı sürekli konuşma tanıma (GDSKT) deneyleri gerçekleştirilmiştir. Türkçe çok eklemeli bir dil olduğundan dolayı, dil modelinin nasıl oluşturulduğu tanıma başarımını ciddi şekilde etkilemektedir. Geleneksel kelime tabanlı dil modelleri İngilizce için iyi sonuçlar verirken, Türkçe'de, Türkçe'nin eklemeli yapısından ötürü, iyi çalışmamaktadırlar. Bu problemle başa çıkabilmek için, genel olarak morfem ve kök-kök sonrası kelime altı birimlerine dayanan, farklı kelime modelleme stratejileri önerilmiştir. Bu çalışmada eğitmensiz şekilde elde edilmiş kelime altı birimler incelenmiştir. Farklı yaklaşımlarla elde edilen bölütlemeler konuşma tanıma aşamasındaki performansları dikkate alınarak karşılaştırılmıştır. En iyi kelime hata oranı 25.24 olarak elde edilmiştir. Aynı deneyde kelime tabanlı dil modelleri kullanıldığında ise kelime hata oranı 26.90 olarak elde edilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

a_{ij}	Transition probability from state i to state j
$C(.)$	Counting function
C_{ij}	Category of the ith morph of the jth word
$\mathcal{M}$	Segmentation model of language
O	Observation sequence
o_i	Observation vectors
W	Word sequence
$\hat{W}$	Estimation of word sequence
w_i	Words in W
μ	Morph

ASR	Automatic Speech Recognition
FSA	Finite State Automaton
FST	Finite State Transducer
HMM	Hidden Markov Model
LM	Language Model
LVCSR	Large Vocabulary Conitnuous Speech Recgnition
MAP	Maximum A Posteriori
MDL	Minimum Description Length
ML	Maximum Likelihood
NMW	Number of Morphs per Word
NON	Non-morpheme
OOV	Out of Vocabulary
PRE	Prefix
STM	Stem
SUF	Suffix
WER	Word Error Rate

1. INTRODUCTION

Automatic Speech Recognition (ASR) is one of the most inspiring research areas in today's world since the road to the era in which humans can talk with machines starts from that point. Before starting to communicate with the machines, we must design these machines such that they can easily (and without errors) transform our utterances to a form which they can gather information, to text. After obtaining the text, gathering the information from that text, using the information and making suitable reactions are other research interests. In other words, in order to live in a world of a science-fiction movie, we must first make the machines recognize our speech.

Although, it is not yet achieved to make the machines recognize our speech enough to understand all the things we say; under some constraints, speech recognition systems show sufficient performance making them usable in daily life. Especially in narrow discourse domains like ticket reservation systems, speech recognition systems do the work making the humans out of use. But this is not yet the case for wider discourse domains. The main difference affecting the performance here is the vocabulary size, i.e., number of distinct words that are needed to be recognized. Also the characteristics of the speech affects the performance critically. Accomplishing the goal where the vocabulary size is large and the speech is as fluent as it is in normal conversation is much harder. Speech recognition systems which are designed for such enviroments are called *Large Vocabulary Continuous Speech Recognition (LVCSR) Systems.*

1.1. Modeling the Language

When vocabulary is large enough it becomes more important how the regularities of the language are modeled in the system. The language models help the recognizer to choose the best sequence giving information about how probable is a word, or basically recognition unit, to follow others.

Language models are generally built using words as the base units and this ap-

proach is suitable for languages like English; however for agglutinative languages like Turkish, this approach does not seem to be appropriate. In the latter case the inflectional nature of the language makes the vocabulary size much larger and that makes the number of words that is not included in the vocabulary of the recognizer, i.e., *out of vocabulary* (OOV) words even larger. Consequently the performance of the system decreases.

Also free word order in Turkish makes the language modeling even challenging. Same sentence can be told with many different orders just to emphasize the meaning.

In order to make language modeling better in such languages, it is suggested to build the language models using sub-word units like morphemes. This new approach to the language modeling brings another research subject within: how to obtain these sub-word units.

1.2. Deriving Sub-word Units

The smallest part of a language that bears a meaning is called a morpheme. How to build words combining the morphemes is bound to morphological rules in languages. Knowing these rules, one can speak using words which are built upon morphemes and analyse the words. Also we can get machines to make this analysis if we model these rules.

At the other hand, since the language is based these basic structures, language should represent some regularities about the morphological structure itself. Utilizing statistical learning methods, we can gather information about the morphological structure of languages. We investigate the subwords that are obtained in such a way in this thesis. Our aim is to examine the performance of the LVCSR systems using these subword units as the basic recognition units of the language model.

2. LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION

Automatic speech recognition (ASR) is one of the most important research areas of natural language processing and also a basic step in many other natural language processing applications. The main goal of ASR systems is to map from an acoustic signal to a string of words. However the characteristics of such systems vary due to the aim of the usage and some external parameters [1].

For applications in which the vocabulary size, i.e., the number of distinct words that are needed to recognize, is small, simple ASR systems, in which even the modeling strategy is easier, would be enough. Another parameter that is needed to be taken into account while designing such systems is how fluent, natural or conversational the speech is. Whereas in applications where the spoken words are surrounded by some sort of pause (isolated speech recognition), the recognition is rather easy; much more complicated systems are needed to recognize continuous speech. There are more parameters affecting the ASR system such as the channel, noise and speaker characteristics.

The ASR system we deal in this thesis is a Large Vocabulary Continuous Speech Recognition (LVCSR) system, in which the speech is continuous and the vocabulary is very large. This type of recognition is the most challenging one and needs the most complicated models. Large vocabulary typically means that the vocabulary, that is used in the application, is roughly larger than 20,000 words.

2.1. Architecture

As it was mentioned, the aim is to convert the acoustic signals into a string of words, i.e. a sentence. If we think the acoustic signals as noisy versions of the sentences, it would be the task of searching for the sentence which has the highest probability of generating the noisy signal, i.e., the acoustic signal, among all possible sentences after modeling the noisy channel. Using this paradigm (noisy channel model), we can

rewrite the ASR problem:

"What is the most likely sentence out of all sentences in the language $\mathcal{L}$ given some acoustic input O?"

In a mathematical notation, the problem is to find the word string, $\hat{W}$ that maximizes $P(W|O)$ where the W is the word string, $w_1, w_2, w_3, ..., w_n$, and the O is a sequence of symbols that represents the acoustic signal, i.e. feature vectors;

$$O = o_1, o_2, o_3, ..., o_t \tag{2.1}$$

$$W = w_1, w_2, w_3, ..., w_n \tag{2.2}$$

$$\hat{W} = \arg\max_{W \in \mathcal{L}} P(W|O) \tag{2.3}$$

After applying the Bayes' rule to Eq. 2.3.

$$\hat{W} = \arg\max_{W \in \mathcal{L}} \frac{P(O|W)P(W)}{P(O)} \tag{2.4}$$

Since maximization is made over possible word strings, i.e., $W \in \mathcal{L}$, for the same observation sequence, the probability of the observation, $P(O)$ is constant and can be neglected. Thus Eq. 2.4 reduces to:

$$\hat{W} = \arg\max_{W \in \mathcal{L}} P(O|W)P(W) \tag{2.5}$$

Eq. 2.5 represents the problem of speech recognition and can be computed by just taking the product of two terms, which are the *prior probability*, $P(W)$ and the *observation likelihood*, $P(O|W)$. The prior probability, $P(W)$ is computed by language models and the observation likelihood,$P(O|W)$ is computed by acoustic models. These models will be explained in following sections.

Adding the feature extraction block, where the acoustic feature vectors extracted from acoustic signals, and the decoding block which searches among the possible word sequences and finds the optimal one; an ASR system can be shown as the block diagram in Figure 2.1.

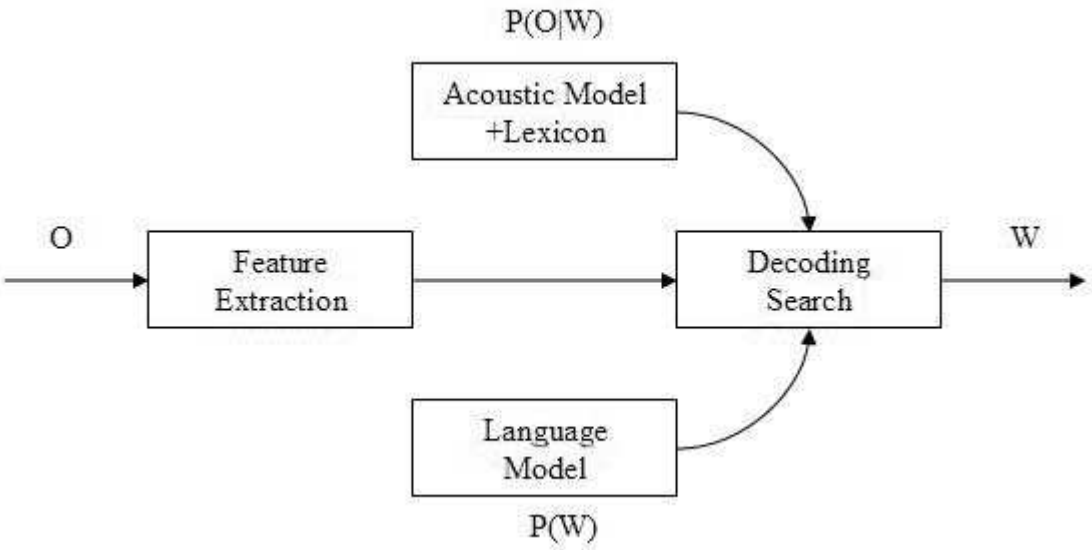

Figure 2.1. Block diagram of an ASR system

2.2. Acoustic Modeling

Observation likelihood $P(O|W)$ is computed using acoustic models. Generally Hidden Markov Models (HMMs) [2] are used for acoustic modeling of which the hidden states are words or subword units like phones or triphones according to the application, and the observations are the observation sequences, i.e. O. Word strings are constructed by concatenating these HMM models of which the states are subword units. In contrary to general HMM structures, HMMs used in speech recognition place strong constraints on transitions; states can only transition to themselves or to succesive states. HMMs that have such a structure are called *Bakis network*.

In LVCSR applications, generally triphones are used in which the effect of neighbouring phones are taken into consideration. Thus the same phones can be represented by different triphones in different contexts.

In order to capture the non-homogeneous nature of phones over time, more than

one state is used to model a phone or a triphone: beginning, middle and finishing states. Adding the non-emitting start and end states, this 5-state HMM structure is called a phone model. Structure of the phone model is shown in Figure 2.2.

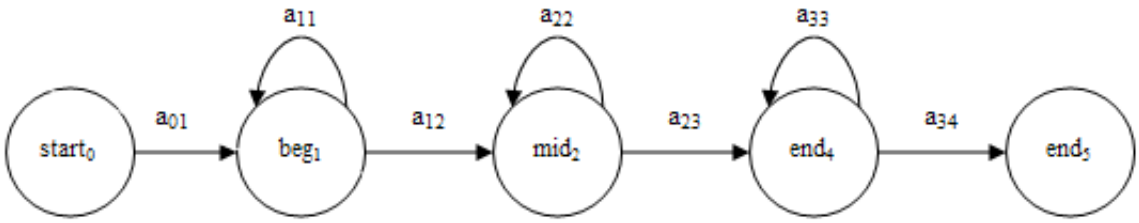

Figure 2.2. Phone model

It must be also mentioned that the observation likelihoods of the HMMs are generally represented as Gaussian Mixture Models in LVCSR applications.

2.3. Language Modeling

Statistical Language Models, which aim to estimate the distributions of natural languages, are essential parts of many natural language processing applications [3]. The estimation process is typically based on just written texts, so is dependent on the training data.

In ASR, language models are used to compute the prior probability, i.e., $P(W)$ which is the second part of Eq. 2.5. It is merely the probability of occurence of a particular word string W.

$$P(W) = P(w_1, w_2, w_3, ..., w_n) \tag{2.6}$$

The chain rule of probability can be used to decompose Eq. 2.6 as follows:

$$P(W) = P(w_1)P(w_2|w_1)P(w_3|w_1^2)...P(w_n|w_1^n) = \prod_{k=1}^{n} P(w_k|w_1^{k-1}) \tag{2.7}$$

where w_n^m stands for the word string $w_n, w_{n+1}, ..., w_{k-1}, w_k$. $P(w_k|w_1^{k-1})$ is the proba-

bility of encountering the word w_k following the word string w_1^{k-1}, which is called the *history*.

As was mentioned, the language models are estimated from written text data, the estimation of the value of $P(w_k|w_1^{k-1})$ would need the count of the word strings, w_1^{k-1} and w_1^k. There is no way of computing these values even if we have a corpus of enormous size. So it is obligatory to make a simplifying assumption: using a limited-sized history, i.e.;

$$P(w_n|w_1^{n-1}) \approx P(w_n|w_{n-N+1}^{n-1}) \tag{2.8}$$

If this limitation allows only previous N-1 words to be taken as the history, *N-gram language models* are obtained. Looking at just the preceding words would result in bigram models, whereas looking at the preceding two would result in trigram models. The prior probability of our essential equation, Eq. 2.5, i.e., $P(W)$ is computed as follows using N-gram models.

$$P(W) = \prod_{i=1}^{n} P(w_i|w_{i-N+1}^{i-1}) \tag{2.9}$$

Estimation of N-gram models is merely a counting and normalization procedure on the training data, i.e., corpus. Basically the number of occurences of the particular N-gram is normalized with the number of occurences of all N-grams with the same N-1 previous words, that is the number of occurences of the word string consisting of N-1 previous words.

$$P(w_n|w_{n-N+1}^{n-1}) = \frac{C(w_{n-N+1}^{n-1}w_n)}{C(w_{n_N+1}^{n-1})} \tag{2.10}$$

Since the natural languages are very generative, language data is sparse even if the size of the data is very large. N-gram models are under curse of this sparseness. It is probable to estimate zero probabilities for N-grams even in low-order models like bigrams since it's normal not to encounter particular word strings in the training data. To overcome this problem, smoothing techniques are introduced where non-zero

probabilities are assigned for unseen data by looking at the low-probability N-grams [1].

2.3.1. Class-based N-grams

Since similiar words can generally be used in similiar context in languages, utilizing classification in language modeling may be useful. Basically, in a class-based fashion, N-gram probabilities are computed using two factors: the probability of a class given the preceding class and the probability of a particular word given the class [3]:

$$P(w_n|w_{n-N+1}^{n-1}) = P(w_n|c_n)P(c_n|c_{n-N+1}^{n-1}) \tag{2.11}$$

These factors can be computed using maximum likelihood estimation as follows:

$$P(w|c) = \frac{C(w,c)}{C(c)} \tag{2.12}$$

$$P(c_i|c_{i-1}) = \frac{C(c_{i-1}c_i)}{\sum_c C(c_{i-1}c)} \tag{2.13}$$

The quality of the models is directly related to classification algorithms that are used. It is shown that manually clustering of semantic categories improves the performance in narrow discourse domains [4], whereas no satisfactory improvement is achieved in larger domains. Automatic clustering algorithms based on information theory also improves the performance after combining the model with its word-based counterpart [5][6].

2.3.2. Evaluation of Language Models

The criteria used in evaluating speech recognition tests can be used to make a comparison between the performance of language models. But, in cases where the

speech recognition is excluded another useful criteria called *perplexity* is generally used. It is an average branching factor for the language model, that is, the higher the perplexity, the larger number of words is taken into account by the recognizer for choosing the following word after recognizing the previous ones. So language models with lower perplexity values are expected to perform better, but it is not always the case in speech recognition tasks.

Perplexity can be computed by 2^H where the H represents the *entropy*, which is a measure of information describing the uncertainty about an event. The entropy on a per word basis can be defined as follows;

$$H(W) = \lim_{n\to\infty} \frac{1}{n} \log P(w_1, w_2, ..., w_n) \tag{2.14}$$

and can be estimated using Eq. 2.15.

$$\hat{H}(W) = \frac{1}{n} \sum_{i=1}^{n} P(w_i | w_{i-N+1}^{i-1}) \tag{2.15}$$

Also the coverage of the test data can be used as an evaluation metric for language models.

2.4. Evaluation of Speech Recognition

Word Error Rate (WER), which is based on how much the word string returned by the recognizer (hypothesized word string) differs from the reference transcription, is the standard evaluation metric for speech recognition systems. In order to find the optimal representation of the hypothesized word string which consists of minimum number of word substitutions, word insertions and word deletions, the minimum edit distance in words between the hypothesized and the reference strings is computed first.

Then the WER is defined as follows:

$$\text{WER} = 100 \cdot \frac{\text{Insertions} + \text{Substitutions} + \text{Deletions}}{\text{Total Words in the Reference Transcription}} \tag{2.16}$$

3. MORPHOLOGICAL ANALYSIS

Morphemes are the smallest meaning-bearing units of languages and morphology is the study of the way words are built up from these smaller units. As it was mentioned before, especially for languages that have a rich morphological structure like Turkish, it can be useful to make use of morphology while constructing language models.

In this chapter, a general survey of (mostly) Turkish morphology will be given and morphological parsing methods will be examined as it is our main purpose. Finally, the structure of the Morfessor, which is a state-of-art unsupervised morphology learner, will be outlined with the modifications aiming to take general characteristics of Turkish morphology into account.

3.1. Survey of Turkish Morphology

Turkish is an agglutinative language, in which the words are constructed by concatenating multiple morphemes [7]. Generally morphemes are divided into two groups: stems, which are the main morphemes in the words, and affixes, which give the additional meanings to words [1]. While affixes are further divided into groups called prefixes, suffixes, infixes and circumfixes; in Turkish only suffixes, which follow the stems and other preceding suffixes in the word, are used. (Although we can rarely see prefixes, which precede the stems; we can easily classify these words as foreign words and consider them as exceptions [8].)

The suffixes in Turkish can be classified as derivational or inflectional in their functional roles. While derivational suffixes are used to create new lexical units and may even change the grammatical categories of words; inflectional suffixes are used just to mark grammatical notions like number, person and gender (nominal inflection) or to mark tense, aspect, modality and person (verbal inflection) [7]. Derivational and inflectional suffixes can be seen in the examples below.

- derivation:
  üçgen : üç + gen [triangle]
  okur : oku + r [reader]

- nominal inflection:
  evler : ev + ler [houses]

- verbal inflection:
  geldi : gel + di [(he) came]

Another difficulty that is encountered while studying Turkish morphology is the vowel harmony. This rule forces certain vowels in the suffix to be compatible with the last vowel of the stem that it is affixed to. Complex characteristics of Turkish morphology can be sembolized with the following extreme example which means "as if you were of those whom we might consider not converting into an Ottoman".

OSMANLILAŞTIRAMAYABİLECEKLERİMİZDENMİŞSİNİZCESİNE
OSMAN -LI -LAŞ -TIR -AMA -YABİL -ECEK -LER -İMİZ -DEN -MİŞ -SİNİZ -CESİNE

3.2. Morphological Segmentation

Our main goal is to construct language models using morphemes and it is an easy task if we have morpheme-based corpora. However morphemes are subword models and their boundaries are not visible in texts. So, our main problem is to learn these boundaries and rebuild the corpora using morphemes. This parsing procedure can be done using a system which knows the morphological rules. General characteristics of such a system will be given below and after that we will begin to talk about how to overcome this learning problem in an unsupervised manner.

3.2.1. Finite State Morphological Parsing

A morphological parser needs to model the following issues in order to achieve morphological segmentation [1].

lexicon: a list of morphemes with the basic information about them (noun, verb, etc.).
morphotactics: a model telling which classes of morphemes can follow other classes.
orthographic rules: spelling rules which models the spelling changes that occurs while combining morphemes.

Although it would be the most useful lexicon which lists all possible words of language, it's impossible to create such a list for various reasons. So computational lexicons usually consists of a list of morphemes with a representation of morphotactics which tells us how to combine them in order to build the words. Although morphotactics can be modeled using many ways, the most common way is the finite-state automaton [1] which is also used in the parser we used in our experiments [9]. A simple FSA that models Turkish nominal inflection is shown in Figure 3.1 in which the nouns are divided into two groups because of vowel harmony rule.

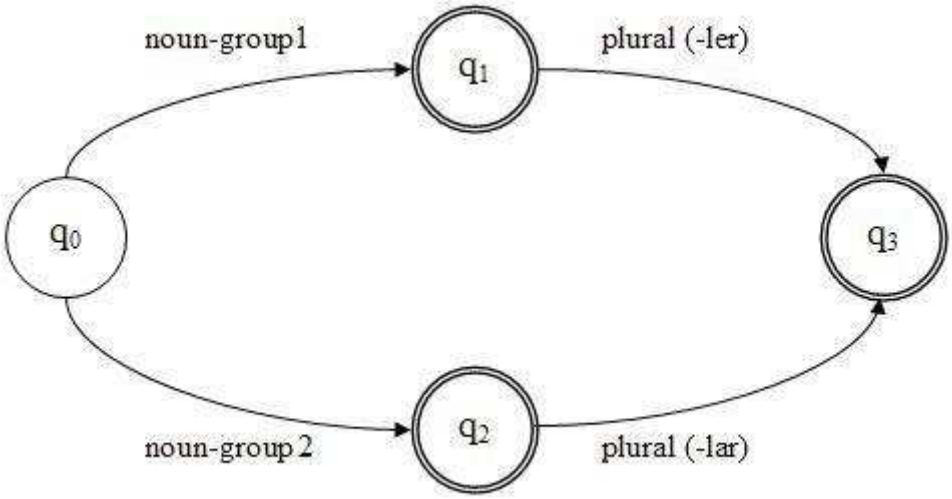

Figure 3.1. A FSA for Turkish nominal inflection

Although using these FSAs can solve the problem of morphological recognition; the problem of determining whether an input string of letters makes up a legitimate

Turkish word or not, in order to solve the parsing problem, some kind of transformation must be utilized. For example, given the input evler (houses), we'd like to output ev +N +PL, telling us evler is a plural noun. We want the parser to transform the word in the surface level, which represents the actual spelling of the word (evler), to a representation in lexical level which gives the concatenation of the underlying morphemes (ev +N +PL). The correspondence between these levels is known as two level morphology [10].

Finite-state transducers, which maps between one set of symbols and another via a finite state automation, are the perfect tools to make such a transformation between word strings. FSTs also can be viewed as machines that read one string and generates another. A transducer for Turkish nominal inflection can be seen in Figure 3.2.

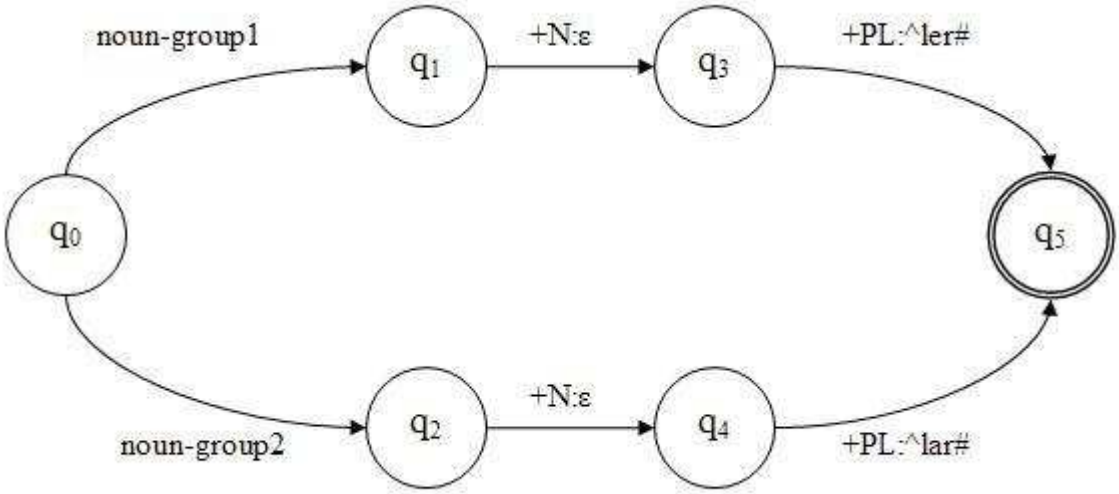

Figure 3.2. A transducer for Turkish nominal inflection

Finally, there is one more issue that the parser must address; orthographic rules. This problem can be overcome by adding a new intermediate level between lexical and surface levels which represents just the morphemes that the word is built of without the spelling changes that occurs after combining the morphemes. An example that shows all these three levels in the case of a basic spelling change in Turkish, vowel drop, is shown below [8].

Lexical: masa +N +1PS +POSSES

Intermediate: masa +Hm
Surface: masam

The transformation between the surface and intermediate levels can be done by another block consisting a series of FSTs modeling the orthographic rules and cascading these two blocks will make the morphologic parser built.

One more problem about the parser would be the ambiguity. For example when trying to parse the word "yüz", the parser is not capable of knowing whether even it is verb (to swim) or noun (face or a hundred). Disambiguation will require some external evidence such as the surrounding words. But this problem is not in the scope of this work.

The outputs of the parser we used for the input "çocukları" are given below.

çocuk[N] +lAr[A3PL] +sH[P3PL] +[NOM] (their children)
çocuk[N] +lAr[A3PL] +sH[P3SG] +[NOM] (his/her children)
çocuk[N] +lAr[A3PL] +[PNON] +YH[ACC] ([I saw] the children)
çocuk[N] +[A3SG] +lArH[P3PL] +[NOM] (their child)

3.2.2. Unsupervised Discovery of Morphemes

Since the morphemes are the basic units of a language, it is evident that a very large corpus would have regularities about them and it is probable to even discover the morpheme boundaries by utilizing basic statistical learning methods. Before beginning to examine these methods, it is obligatory to make the definition of the term; morph. Since the subword units that are discovered by an unsupervised manner wouldn't be actual morphemes, a new term, morph, is used to define them [11].

The first basic strategy about finding morpheme boundaries was proposed by Harris in 1955 [12]. The strategy was based on the suggestion that the morpheme boundaries should be at locations where the predictibility of the next letter of a letter

sequence is low. Various following work on the area was motivated by that basic idea. Hafer and Weiss improved the algorithm while identifying it with a probabilistic notion and also proposed a basic measure [13]. Dejean tried to list the frequent suffixes of a language instead of segmenting all the words in the corpus using Finite State Automaton (FSA) based on the same idea [14]. Also neural networks were utilized by Elman using the same idea [15].

Statistical learning methods like Maximum Likelihood (ML), Minimum Description Length (MDL) and Maximum a Posteriori (MAP) based modeling are widely used in formulating segmentation algorithms (As it is the case in Morfessor). While in ML modeling, additional heuristics are needed in order not to overlearn; models based on MDL and MAP considers not only the model accuracy but also the model complexity and don't need any aditional heuristics.

Segmenting texts in which there are no word boundaries to words is another research area and is similiar to our case. This process is the first step of any natural language processing application in languages where the word boundaries are not used like Chinese and Japanese. de Marcken generally deals with that problem in his thesis using MDL-based modeling [16]. But the algorithm doesn't give satisfactory results when used for morpheme segmentation.

Many existing morphology discovery algorithms assume a rather simple morphology and usually try to split the words into just two parts. This approach is not suitible for agglutinative languages like Turkish. Goldsmith exceeds this limitation using a recursive structure, where it's allowed for stems to have inner structures, in his MDL-based morphological analyser, Linguistica [17].

3.3. Morfessor

In contrast with many of other unsupervised morpheme discovery algorithms, Morfessor - another unsupervised morphology learner and morpheme segmenter developed by Mathias Creutz [18] - assumes multiple segmentations per word and is more

appropriate for agglutinative languages like Turkish. Also it's shown that it outperforms the previous algorithms in the area [18]. A Bayesian framework is used while developing the Morfessor; thus, it can be said that the resulting lexicon should cover the training corpus well and additionally generalize to new word forms not observed in the training data. Since Morfessor is mainly used in this work, we will examine the development steps and the mathematical basis of the algorithm below.

3.3.1. Development Steps of Morfessor

The Morfessor model has undergone four development steps while trying to avoid under and oversegmentation as well as morphotactic violations. These categories are called; baseline, baseline-freq-length, categories-ML and categories-MAP.

Baseline model, as the name suggests, is the simplest and the one that is used as bootstrapping step by other models. The model is based on a two-part-code crude MDL [11] and also an equivalent MAP formulation for the model can be given [19].

Applying Bayesian prior probabilities to the frequency and length distributions of the morphs to the baseline model creates the second model, baseline-freq-length model which outperforms the preceding model by reducing the number of under and oversegmented words [20]. But still the model is insufficient about morphotactic violations. The prior probability distributions that are used in the model will be given in the mathematical formulation.

The errors caused by context-insensivity in the baseline models is reduced by applying simple morphotactics in the categories-ML model [21]. Each morph segmented by the baseline model is tagged with one of four morph categories; prefix, stem, suffix or noise, using some usage-based features of the morphs. After this tagging procedure, the segmentation is reanalized using maximum likelihood (ML) maximization. The morphs that are tagged as noise are concatenated to their neighbours to reduce the oversegmentation. A first order Hidden Markov Model (HMM) is used for assigning probabilities to the segmentations in which there are transition probabilities between

categories and emmision probabilites indicating how likely a morph is to occur given a category. The HMM models a simple morphotactics that is shown by the following regular expression:

word = (prefix* stem suffix*)+

The aforesaid morphotactics is changed in order to model the Turkish morphology better by avoiding prefixes. Modified regular expression is given below.

word = (stem+ suffix*)+

In the final and the most sophisticated Morfessor model, Categories-MAP, the algorithm operates on word-tokens, whereas Categories-ML operates on word types, and thus can utilize the frequency of the words which is a valuable information [22]. Also the new model is a complete maximum a posteriori model in which there is no need to rely on additional heuristics to determine the optimal lexicon size. Furthermore hand the model has a hierarchical lexicon structure which allows the lexicon to be coded with a lower cost. In such a lexicon, each morph in the lexicon can consist of two submorphs which are also present in the lexicon and the noise-categorized morphs are allowed in order to reduce the cost whereas the appropriate segmentation is also represented in the lexicon.

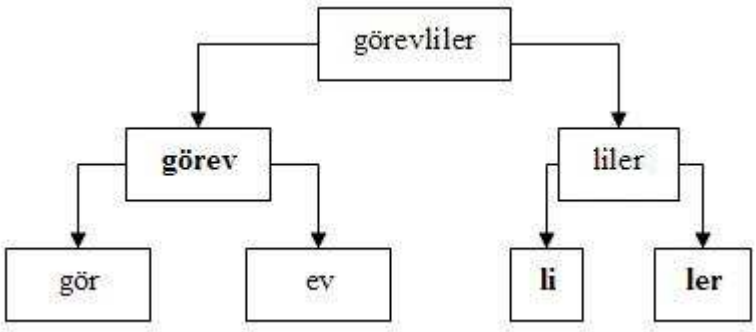

Figure 3.3. Hierarchical representation of the word “görevliler”

3.3.2. Mathematical Formulation of Morfessor

The aim of the Morfessor is to induce a model of the language using just a corpus in an unsupervised manner. The model of the language ($\mathcal{M}$) includes a morph lexicon and a grammar determining how the words are built using the morphs. (Baseline models just tries to induce the lexicon without the grammar.) Since the task is to find the optimal segmentation, it can be achieved through maximum a posteriori (MAP) optimization by maximizing $P(\mathcal{M}|corpus)$ [23]. After applying Bayes' rule;

$$\arg\max_{\mathcal{M}} P(\mathcal{M}|corpus) = \arg\max_{\mathcal{M}} P(corpus|\mathcal{M}) \cdot P(\mathcal{M}) \tag{3.1}$$

where the probability of the model is the joint probability of the probabilities of the induced lexicon and grammar.

$$P(\mathcal{M}) = P(lexicon, grammar) \tag{3.2}$$

3.3.2.1. Lexicon. Probability of constructing a lexicon which includes M distinct morphs could be given as;

$$P(lexicon) = P(size(lexicon) = M) \cdot P(properties(\mu_1), ..., properties(\mu_M)) \cdot M! \tag{3.3}$$

The prior probability that the lexicon contains exactly M morphs can be neglected since that value is of no practical significance in the computation of a model involving thousands of morphs.

The properties of morphs can be divided into two groups about their usage and form which will be described afterwards.

$$P(properties(\mu_i)) = P(usage(\mu_i), form(\mu_i)) \tag{3.4}$$

3.3.2.2. Grammar. Since in the baseline models there is no categorization, there is no knowledge about how the morphemes are combined. Thus equation 3.2 is reduced to:

$$P(\mathcal{M})=P(lexicon) \tag{3.5}$$

In categories-ML and categories-MAP models, the grammar is represented by a simple morphotactics realized as a Hidden Markov Model (HMM) of which the states correspond to grammatical categories: prefix(PRE), stem(STM), suffix(SUF) and non-morpheme(NON). Nevertheless, since it is convenient to use uniform transition probabilities, $P(grammar)$ is asseumed as a constant. Thus the $P(\mathcal{M})$ in equation 3.2 is reduced to $P(lexicon) \cdot k$.

To utilize the HMM in the segmentation procedure, we need not only the transition probabilities but also the emission probabilities $P(\mu_i|C_i)$. Using Bayes' rule;

$$P(\mu_i|C_i) = \frac{P(C_i|\mu_i) \cdot P(\mu_i)}{P(C_i)} = \frac{P(C_i|\mu_i) \cdot P(\mu_i)}{\sum_{\forall \mu_{i'}} P(C_i|\mu_{i'}) \cdot P(\mu_{i'})} \tag{3.6}$$

where the probability of a morph to be assigned to a particular category $P(C_i|\mu_i)$ is derived using the usage-related parameters of the morph which will be explained afterwards.

$$P(C_i|\mu_i) = P(C_i|usage(\mu_i)) \tag{3.7}$$

Also it must be pointed out that the category-independent probabilities of the morphs, $P(\mu_i)$, are just the maximum likelihood estimates.

3.3.2.3. Corpus. After inducing a particular morph segmentation, it is easy to calculate the probability of the corpus given the model of the language.

$$P(corpus|\mathcal{M}) = \prod_{j=i}^{W} \Big[P(C_{j1}|C_{j0}) \prod_{k=1}^{n_j} [P(\mu_{jk}|C_{jk}) \cdot P(Cj(k+1)|Cjk)] \Big] \tag{3.8}$$

where all the words in the corpus, wihch are each segmented into n_j morphs, are taken into account. A representative HMM is shown in Figure 3.4.

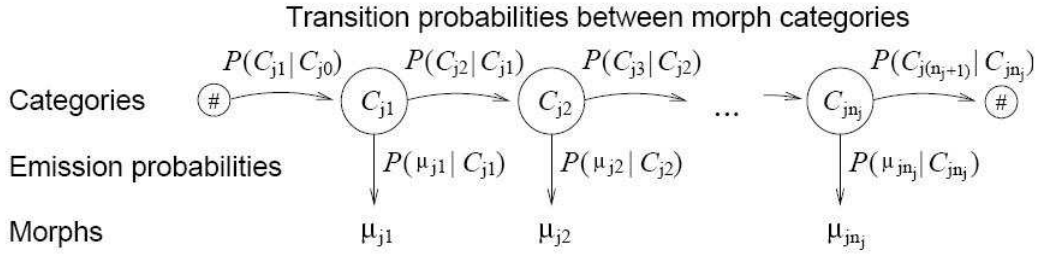

Figure 3.4. HMM representing the structure in Categories Models [18].

3.3.2.4. Properties of Morphs. As it is mentioned above, the properties of morphs can be divided into usage-based and form-based features. While the form of a morph is just about the representation of the morph, the usage-based features are about the properties of the morph and the context that the morph appears in. So we can assume these two group of features as independent features.

$$P(properties(\mu_1), ..., properties(\mu_M)) = \\ P(usage(\mu_1), ..., usage(\mu_M)) \cdot P(form(\mu_1), ..., form(\mu_M)) \tag{3.9}$$

3.3.2.5. Form of Morphs. A morph is represented either by a string of letters or by its substructure since a hierarchical structure is used in the lexicon. Thus the form of each morph can be assumed as independent from the others.

$$P(form(\mu_1), ..., form(\mu_M)) = \prod_{i=1}^{M} P(form(\mu_i)) \tag{3.10}$$

Probability of the form of a morph can be calculated by;

$$P(form(\mu_i)) = \begin{cases} (1 - P(sub)) \cdot \prod_{j=1}^{length(\mu_i)} P(c_{ij}) \\ P(sub) \cdot P(C_{i1}|sub) \cdot P(\mu_{i1}|C_{i1}) \cdot P(C_{i2}|C_{i1}) \cdot P(\mu_{i2}|C_{i2}) \end{cases} \tag{3.11}$$

where $P(sub)$ is the probability that a morph has substructure and is just a maximum likelihood estimation.

While, the first part of equation 3.11 is used for the morphs that doesn't have substructure and the product is taken over the probabilities of the letters, c_{ij}, that the morph consists of; the second part is used for the morphs that is a concatenation of two submorphs. The probabilities of the letters are basically estimated from the corpus. Since only the Categories-MAP model has hierarchical structure, $P(sub)$ is zero for other models.

3.3.2.6. Usage Related Features. Although a very large set of features can be used as usage related features of a morph, a limited set of features are used in Morfessor. The *frequency* of the morph in the segmented corpus and the *length* of the morph in letters are used as the features of the morph itself. Also intra-word *right* and *left perplexity* of the morph are used as context dependent features of the morph. Thus, using these properties, the probability of the usages of the morphs in the lexicon becomes:

$$P(usage(\mu_1), ..., usage(\mu_M)) = P(freq(\mu_1), ..., freq(\mu_M)) \cdot \prod_{i=1}^{M} [P(length(\mu_i)) \cdot P(right - ppl(\mu_i)) \cdot P(left - ppl(\mu_i))] \quad (3.12)$$

While the frequencies of the morphs are given as a joint probability, other usage related probabilities are taken as independent of the corresponding values of other morphs due to practical considerations.

3.3.2.7. Frequency. The joint probability of the frequencies of the morphs can be expressed using a non-informative prior where only the morph token and morph type values are taken into account.

$$P(freq(\mu_1), ..., freq(\mu_M)) = 1/\begin{pmatrix} N-1 \\ M-1 \end{pmatrix} = \frac{(M-1)!(N-M)!}{(N-1)!} \quad (3.13)$$

where N is the total number of morph tokens and M is the total number of morph types.

In the Morfessor Baseline-Freq-Length model, a Bayesian frequency prior, which is based on a correction of Zipf's law, is used as the probability of the frequency af a morph.

$$P(freq(\mu_i)) = freq(\mu_i)^{log_2(1-h)} - (freq(\mu_i) + 1)^{log_2(1-h)} \tag{3.14}$$

The parameter h is the prior belief of the proportion of *hapax legomena*, i.e., morph types that are encountered just once in the corpus. Typically, that value is about half of all morph types. Also in this case, the frequency of one morph is assumed to be independent of the frequencies of the others, that is:

$$P(freq(\mu_1), ..., freq(\mu_M)) = \prod_{i=1}^{M} P(freq(\mu_i)) \tag{3.15}$$

Zipf's law is typically the result of his study about the relationship between the frequency of a word and its rank, i.e., the position of the word in a list where the words have been sorted according to falling frequency. Resulting probability distribution in this case, where the h is taken as 0.5, is shown in Figure 3.5 below.

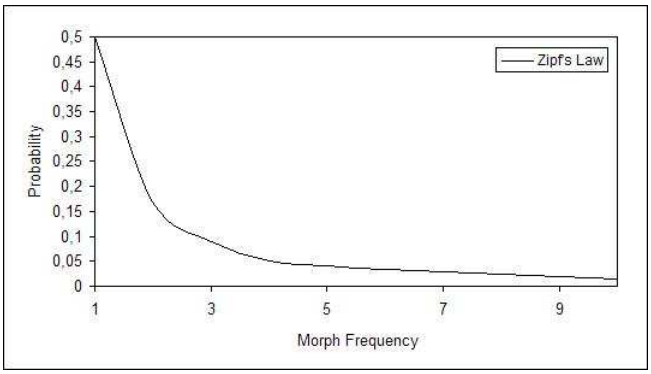

Figure 3.5. Zipf's law

3.3.2.8. Length. The probability that a morph of particular length $length(\mu_i)$ will emerge is:

$$P(length(\mu_i)) = [1 - P(\#)]^{length(\mu_i)} \cdot P(\#) \tag{3.16}$$

where # represents a special end-of-morph character which is used only for this calculation. This value is just the result of encountering $length(\mu_i)$ before encountering the end-of-morph character. According to this distribution, the probability of observing a morph of a particular length decreases exponentially with th length of morph.

The length can be modeled explicitly as a gamma distribution in the Morfessor Baseline-Freq-Length model.

$$P(length(\mu_i)) = \frac{1}{\Gamma(\alpha)\beta^{\alpha}} lenght(\mu_i)^{\alpha-1} e^{-length(\mu_i)/\beta} \tag{3.17}$$

where

$$\Gamma(\alpha) = \int_0^{\infty} z^{\alpha-1} e^{-z} dz \tag{3.18}$$

The constants, α and β determine the exact shape of the curve. The maximum value of the density occurs at $length(\mu_i) = (\alpha - 1)\beta$, which corresponds to the most common morph length in the lexicon.

Information about the length of a morph is also used while assigning category tags to the morphs. It is unlikely for a morph to be a stem when it is not long enough. It is evident that the stems need to be longer to be more distinguishable from each other since the set of stems is very large in natural languages.

3.3.2.9. Intra-Word Right and Left Perplexity. Right and left perplexity of morphs are measurements of how predictible the following and the preceding morph is, respectively. It is reasonable to suggest that it is more difficult to predict what the following morph is going to be after a prefix, that is the right perplexity is high for a prefix, vice versa. So right and left perplexity can be used in the category-tagging procedure.

The right perplexity of a morph can be calculated as follows.

$$right - ppl(\mu_i) = [\prod_{v_j \in right-of(\mu_i)} P(v_j|\mu_i)]^{-\frac{1}{freq(\mu_i)}} \tag{3.19}$$

where the value of $P(v_j|\mu_i)$ is calculated over all v_j, i.e., the number of occurences of morph tokens immediately following the occurences of μ_i.

3.3.2.10. Category Membership Probabilities. As it was mentioned, Morfessor Categories models also assigns categories to morphs. Usage related features of the morphs, which are told to be used in the calculation of the global cost function, can also be used for assigning categories to the morphs. The categories that are used in Morfessor are prefix (PRE), stem (STM), suffix (SUF) and non-morpheme (NON), where the latter is only used for hierarchical representation.

Right and left perplexity of morphs, which are given in the previous section, can be indirectly used in this category-tagging scheme, using a prefix or suffix-likeness function, which are essentially sigmoid functions of right and left perplexity, respectively. The function of $prefix - likeness(\mu_i)$ is given in Equation 3.20, to which the $suffix - likeness(\mu_i)$ is identical except of that the $right - ppl(\mu_i)$ is used instead of $prefix - likeness(\mu_i)$.

$$prefix - like(\mu_i) = (1 + exp[-a \cdot (right - ppl(\mu_i) - b)])^{-1} \tag{3.20}$$

Analogously, a $stem - likeness$ function can be constructed using the length of morphs instead of perplexity values in the preceding function. It's explained above how the length of morphs gives information about stem-likeness of a morph.

$$stem - like(\mu_i) = (1 + exp[-c \cdot (length(\mu_i) - d)])^{-1} \tag{3.21}$$

While, the parameter b Eq. 3.20 is the threshold indicating the point where the

morph is as likely to be a prefix as a non-prefix; the parameter d in Eq. 3.21 is the threshold of stem-likeness. The parameters a in Eq. 3.20 and c in Eq. 3.21 govern the steepness of the sigmoids. The sigmoid functions are shown in Figure 3.6.

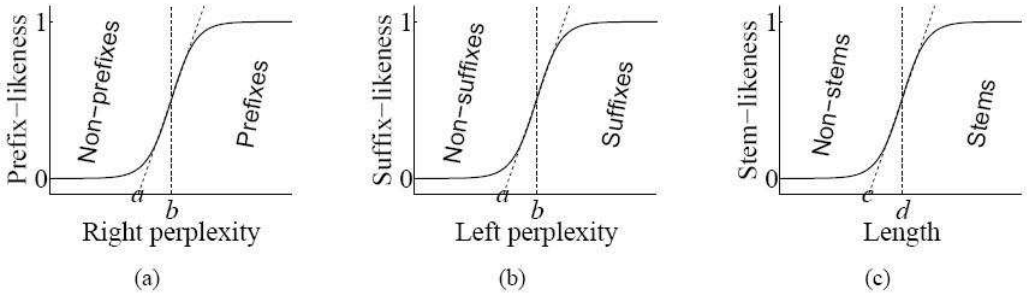

Figure 3.6. Sigmoid functions representing (a) prefix-likeness, (b) suffix-likeness and (c) stem-likeness [18].

Although the functions aforesaid assumes values between zero and one, they are not actual probability values since they do not sum up to one. Also there is a fourth category, NON, which should be introduced. The segments that are not long enough to be stems and that have low right and left perplexity values are tagged as NON. An appropriate equation which can be used to calculate the probability of a morpheme to be a NON is given below.

$$P(NON|\mu_i) = [1 - prefix - like(\mu_i)] \cdot [1 - psuffix - like(\mu_i)] \cdot [1 - stem - like(\mu_i)] \tag{3.22}$$

Probabilities of the other tags can be obtained by distributing the remaining probability mass. The probability that a segment is a PRE is given in Eq. 3.23 where the others can be obtained analogously. The exponent q in the equation affects the normalization where lower values makes the distributions flatter.

$$P(PRE|\mu_i) = \frac{prefix - like(\mu_i)^q \cdot [1 - P(NON|\mu_i)]}{prefix - like(\mu_i)^q + stem - like(\mu_i)^q + suffix - like(\mu_i)^q} \tag{3.23}$$

As it is mentioned, we made a modification to the algorithm to make it more appropriate for Turkish by not allowing prefixes. In that case the equation that gives the probability that a segment is a SUF reduces to Eq. 3.24 which is also an appropriate

probability definition.

$$P(SUF|\mu_i) = \frac{suffix - like(\mu_i)^q \cdot [1 - P(NON|\mu_i)]}{stem - like(\mu_i)^q + suffix - like(\mu_i)^q} \tag{3.24}$$

3.3.3. Search Heuristics

The mathematical formulation of Morfessor, which is explained in details in the previous section, provides the way to calculate an overall probability for every segmentation of the corpus. But the aim of the model is to find the segmentation and the lexicon that gives the highest probability and since the search space is all the possible segmentations of the given corpus, without a good search algorithm the model would be useless.

First, the search algorithm that is used in the Baseline model must be explained since it is used as a bootstrapping step in the other models. In that model, the list of all the words in the corpus is taken as the morph lexicon and the overall cost is calculated. After that, different morph segmentations are proposed in each step and the cost is calculated again. Segmentations increasing the overall probability are selected and this procedure continues until no significant improvement is obtained.

All the distinct entries of the lexicon are explored to find the optimal split into two substrings in each step. In the case of a split, the splitted parts are again explored in a recursive fashion until there is no more gain of splitting the strings. After processing all the entries in the lexicon, the lexicon - with the new entries - is shuffled randomly and the search continues.

The Baseline model gives the resulting lexicon at this point. But the other models take this lexicon and go on searching. First, all the entries in the lexicon are tagged using the equations that are given in the mathematical formulation section. After that, the possible splits of morphs into two substrings are explored again but by using the tags and taking the context into account this time. After the splitting step,all

the entries are re-tagged using Viterbi algorithm and the next step begins; joining of morphs. All the morph bigrams are searched and the morphs are joined to form a new longer morph if it would be a better segmentation in this step while remaining the hierarchical structure in Categories-MAP model and all the morphs are re-tagged using Viterbi algorithm again. Splitting and joining steps are repeated again and the resulting segmentation structure is expanded to the finest resolution which excludes non-morphemes finally.

3.3.4. Segmenting Corpora With an Existing Model

It's mentioned that the segments that are derived using the Morfessor can be tought as the observation sequences of an HMM in Categories Models. It's also the case for the Baseline Model in which the HMM has only one hidden stage due to the absence of tags. Thus a new corpus can be segmented using an exising model using the well-known Viterbi Algorithm.

3.3.5. Evaluation

Since our aim is to make the investigation from the perspective of speech recognition, standard speech recognition results can be used to evaluate the segmentation experiments. But this approach wouldn't take the actual segmentation into account and wouldn't be a true evaluation for the morphological analysis process. In order to examine how the automatically derived units resembles the actual segmentation we can use precision, recall and F-measure.

Whereas precision is the proportion of correct boundaries among all morph boundaries suggested by the algorithm; the recall is the proportion of correct boundaries discovered by the algorithm in relation to all morpheme boundaries in the actual segmentation.

$$Precision = \frac{C(\text{Correctly Estimated Boundaries})}{C(\text{All of The Estimated Boundaries})} \tag{3.25}$$

$$Recall = \frac{C(\text{Correctly Estimated Boundaries})}{C(\text{Actual Boundaries})} \tag{3.26}$$

F-measure, which is the harmonic mean of the two can be used as the one evaluation metric.

$$F - Measure = 1/[\frac{1}{2}(\frac{1}{Precision} + \frac{1}{Recall})] \tag{3.27}$$

4. EXPERIMENTS

Since the aim of the thesis is to investigate the automatically derived subword units for Turkish LVCSR, the experiments were performed by changing the parameters that affects the usage of the morphs and using different segmentation strategies. In this chapter, after giving general information about the corpus that was used in the experiments, different segmentation schemes will be examined with their results. Language models that are built using the results of different segmentation procedures will be examined. After that we will begin to talk about the recognition experiments. Basic statistics about the speech data will be given and we will finish the chapter with recognition results.

4.1. Morphological Segmentation

The segmentation experiments were performed using the Morfessor after applying a simple modification that was explained in the previous chapter. Outputs of different experiments that were performed using different parameters are examined in this section. Also the language models that are built using these outputs are compared.

4.1.1. The Corpus

Since the segmentation is performed using just a corpus and in an unsupervised manner; larger the corpus we use the better results we would get. The corpus we have used in the segmentation experiments was acquired from web news portals using a spider that is developed at Boğaziçi University Computer Engineering Department by Haşim Sak [9]. This corpus consists of 182,285,763 word tokens and 1,825,083 word types.

Using such a large corpus has its own drawbacks like it's probable to encounter noise in it. Although the words, even the sentences that contains inappropriate characters are deleted, strings of letters that are not words of Turkish are seen in the

corpus like website addresses. Assuming that these noise-strings are very sparse, using words that are seen more than some number in the segmentation process is a probable solution.

After segmenting the corpus with these more frequent words, all the corpus is segmented using the resulting models.

4.1.2. Experiments

As it's mentioned, we've used the words that are encountered more than some number in the corpus in order to cope with the noise of the corpus. We have performed the segmentation experiments taking words that are seen more than 5, 10, 20, 50 and even 100 times in order not only to solve the noise problem, but also see the behaviour of the segmenter about the frequency of the words of the language and we'll call these experiments as segmentation experiment 1, 2, 3, 4 and 5 respectively.

The first experiment has been performed using the words that are encountered in the corpus more than 5 times. The corpus consists of 511,396 distinct words (types) and 180,260,699 word tokens in total after eliminating the rare words. Basic statistics about the result of the experiment is given in Table 4.1 below where the NMW stands for "Number of Morph per Word".

Table 4.1. Statistics about the outputs of the segmentation experiment 1

Stem Type	154,917
Stem Count	182,357,300
Suffix Type	535
Suffix Count	102,955,523
Morph Type	155,452
Morph Count	285,312,823
Mean of NMW	2.28

One of the words that are segmented to the maximum number of morphs in this experiment is "değer/STM +le/SUF +n/SUF +di/SUF +r/SUF +me/SUF +ler/SUF

+im/SUF +iz/SUF +i/SUF +n/SUF" with 11 morphs.

The same statistics after segmenting the whole corpus are shown in Table 4.2 below.

Table 4.2. Statistics about the outputs of the segmentation experiment 1 after Viterbi

Stem Type	263,456
Stem Count	184,920,131
Suffix Type	592
Suffix Count	106,070,781
Morph Type	264,048
Morph Count	290,990,912
Mean of NMW	2.70

Distributions of NMW before and after the Viterbi segmentation is shown in Figure 4.1.

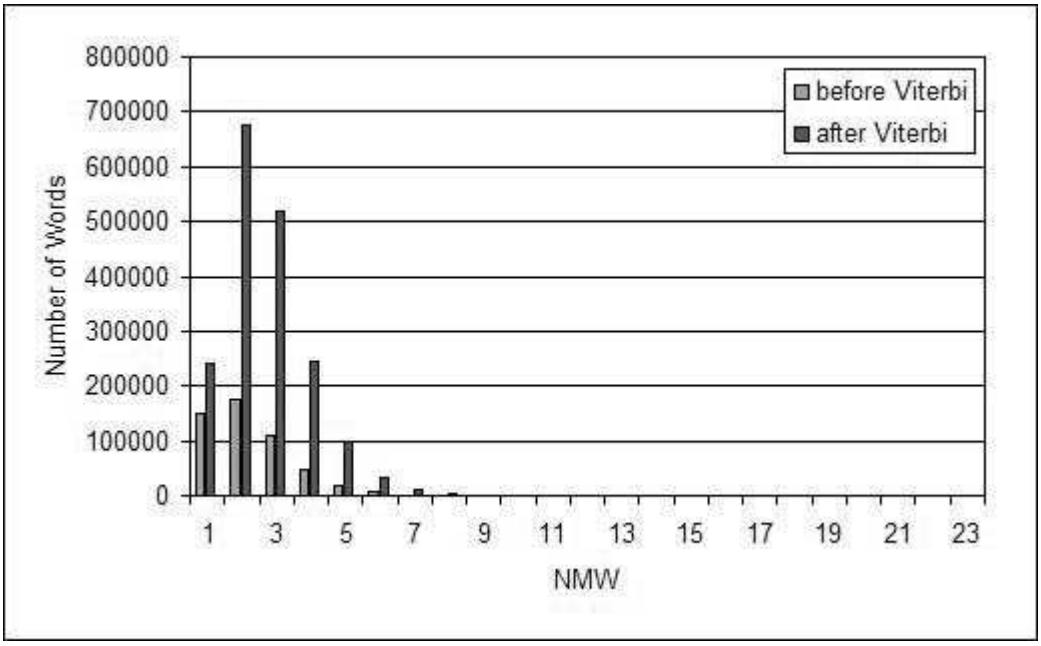

Figure 4.1. Distribution of NMW

The word type and word token counts of the corpora that were used in the other experiments after the elimination process are given in Table 4.3. Also the same basic statistics about the other experiments are given in Table 4.4.

Table 4.3. Statistics about the corpora

	Word Type	Word Token
Experiment 2	340,987	179,145,583
Experiment 3	230,724	177,646,421
Experiment 4	137,031	174,735,505
Experiment 5	90,772	171,490,832

Table 4.4. Statistics about the segmentation experiments

Word Token Threshold	10	20	50	100
Stem Type	262,924	280,063	276,470	316,788
Stem Count	186,538,697	188,391,744	190,029,318	190,965,737
Suffix Type	427	298	207	142
Suffix Count	102,130,928	96,158,479	99,268,056	196,420,553
Morph Type	263,351	280,361	276,677	316,930
Morph Count	288,669,625	284,550,223	289,297,374	287,386,290
Mean of NMW	2.75	2.78	2.93	2.92

NMW distributions of all the remaining experiments is shown in Figure 4.2.

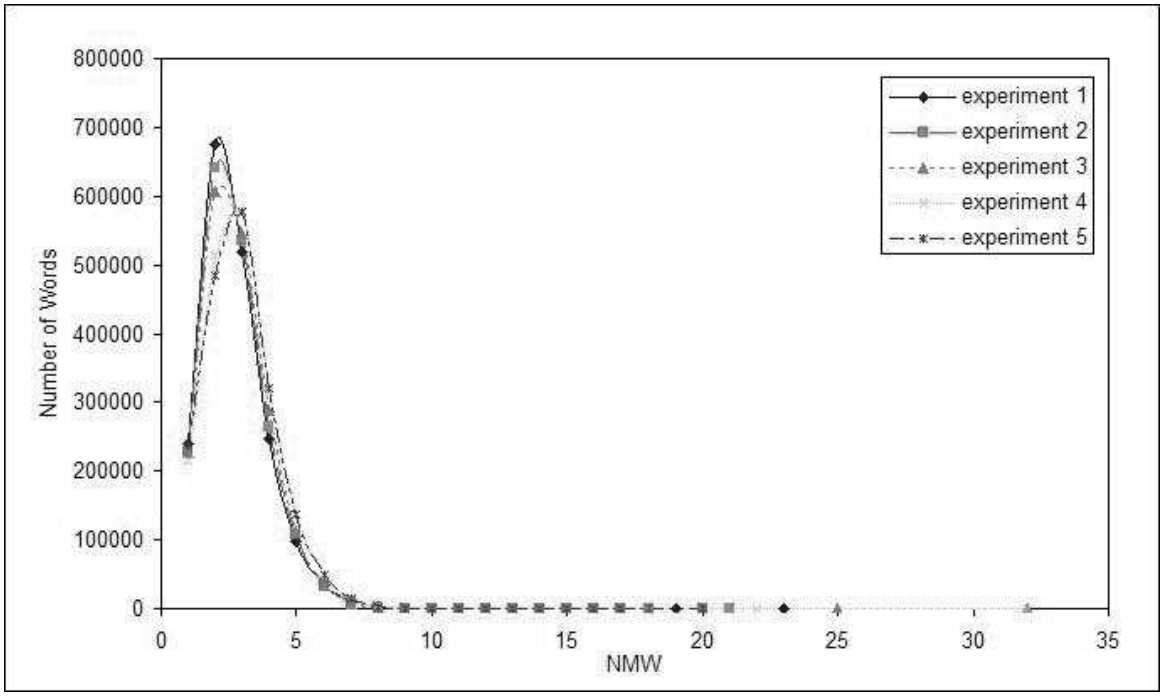

Figure 4.2. Distributions of NMW resulted from five different experiments

After completing the segmentation experiments with five different-sized corpora,

three more experiments were performed using the original Morfessor allowing the morphs to be tagged as prefixes with the words encountered more than 5, 10 and 20 in the corpus. Same statistics that are given in preceding experiments is shown in Table 4.5 where columns 5, 10 and 20 shows the results of the experiments in which the words encountered more than 5, 10 and 20 are used, respectively.

Table 4.5. Statistics about the outputs of the segmentation experiments that were performed using the original Morfessor

Word Token Threshold	5	10	20
Prefix Type	366	219	119
Prefix Count	5,409,668	4,254,033	2,519,408
Stem Type	385,230	284,822	292,695
Stem Count	183,188,224	184,962,349	186,375,320
Suffix Type	568	447	313
Suffix Count	50,263,236	103,153,619	101,834,041
Morph Type	385,798	285,269	293,008
Morph Count	233,451,460	288,115,968	288,209,361
Mean of NMW	2.42	2.78	2.87

Although, a lot of distinct prefixes are induced in the contrary of our belief; we can easily say that most of these prefixes are not present in Turkish morphological system, even for foreign words. For example, the prefix “ab” in the following words are false estimations.

ab/PRE + artacak/STEM +larına/SUF
ab/PRE +anıyor/STEM +lar/SUF

4.1.3. Evaluation

Although our aim is to compare the effect of different segmentation approaches to the speech recognition tasks, segmentation results can be evaluated using precision and recall. The first 5 experiment can be compared in Figure 4.3. Also the segmentation resulted from the Baseline model is evaluated using the same criteria.

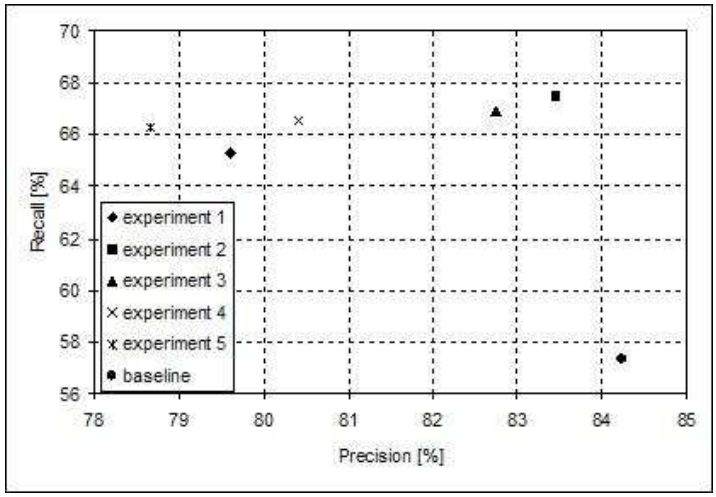

Figure 4.3. Precision and recall

4.2. Recognition Experiments

In each recognition experiment we have used a new language model that is built using the segmented corpora by each of the segmentation experiments. We have also tried different modeling strategies and compared the results. While performing each of these experiments, the same acoustic model was used. Before explaining the experiments, basic statistics about the data that is used while building the acoustic model will be given.

4.2.1. Acoustic Model Data

Speech data are needed to build acoustic models. In our experiments, Turkish Broadcast News database, which has been collected in Boğaziçi University, was used. The database contains daily broadcast news recordings from four different TV channels and a radio channel.

Time analysis of the database with acoustic conditions and channel information is given using Hub4 classes in Table 4.6. These classes are: (f0) clean speech, (f1) spontaneous speech, (f2) telephone speech, (f3) background music, (f4) degraded acoustic conditions and (fx) other.

Table 4.6. Time analysis of the database with acoustic conditions and channel information

Training Data							
Channel	f0	f1	f2	f3	f4	fx	Total
CNN	15.0	7.8	1.9	6.6	22.9	1.0	55.2
NTV	15.5	3.8	2.2	6.8	36.9	1.7	66.9
TRT2	5.1	1.6	0.2	2.7	8.0	0.2	17.8
HI	11.9	0	0	0	0	0	11.9
TRT1	1.2	1.4	0	0.4	2.6	0.1	5.7
VoA	17.0	0.9	4.0	3.0	1.3	0.1	26.3
Total	65.7	15.5	8.3	19.5	71.7	3.1	183.8
Test Data							
CNN	0.13	0.02	0	0.12	0.45	0	0.72
NTV	0.18	0.09	0.03	0.010	0.28	0.01	0.69
TRT2	0.18	0.03	0	0.22	0.46	0	0.89
VoA	0.62	0	0.04	0.08	0.07	0	0.81
Total	1.11	0.14	0.07	0.52	1.26	0.01	3.11

While the acoustic data that is used in previous works on the database consisted of 68.6 hours speech records [24], [25]; the data that is used in our experiments approaches nearly 200 hours with new recordings. While the data that is recorded until April 2007 is used for training, the data that is recorded in April 2007 is used for test. Thus, training and test data don't overlap, even by the month of recording. 11.9 hours data that is tagged as HI in the Table 4.6 is a recorded program called "news for the hearing impaired", which is clearly articulated speech.

4.2.2. Language Modeling

Since the data that is used for language modeling is examined in previous sections and language modeling is explained in details in Chapter 2, it would be enough to talk about the morph structure that is used for the language modeling part.

Two different structure of recognition units based on morphs are used while building the language models: morphs and morph-stem-endings. For the morphs, whole morphological segmentation is represented in the recognition units with adding just "+" to the left of the suffixes in order to seperate them from stems. While the same structure is used for the morph-stem-ending recognition units, suffixes are concatenated. Also it must be emphasized that 4-gram language models are used for the morph-stem-ending based models and 5-gram language models are used for the morph based models.

Morph: açabilir +ler +di
Morph-stem-ending: açabilir +lerdi

Also a different scheme is used while building the language model based on the results of the original Morfessor which allows morphs to be tagged as prefixes. Only difference in this scheme is that "-" signs are added to the prefixes from the right in order to make them recognizable. For the output of the same word;

aça- bilir +ler +di

The language models have been built using the segmentation obtained from the experiments which are explained above. Language models are compared, using the perplexity as the metric, in Table 4.7 where the LM-1, LM-2, LM-3, LM-4 and LM-5 stands for the language models that are built upon the segmentations which are obtained from the segmentation experiment 1, 2, 3, 4 and 5 respectively. The perplexities were normalized using the ratio of morphs to the words to make a fair comparison [26] by the following formula.

$$P_N = P_m^{\frac{\#morphs}{\#words}} \tag{4.1}$$

The P_N stands for the normalized perplexity and the P_m is the perplexity of the morphs.

These ratio values were given in Table 4.2 and Table 4.4 for each segmentation

experiment. And the ratio for the stem-ending models has been found as approximately 1.87. Also it must be emphasized that the language models that have been used in the recognition experiments have been pruned in order to make the recognition experiment feasible due to computational limitations.

Table 4.7. Total log-probabilities and normalized perplexities of the language models

	Based on morphs		Based on stem-endings	
	Log-probability	Perplexity	Log-probability	Perplexity
LM-1	-84563.8	4684.06	-78582.0	2576.16
LM-2	-83874.8	4372.35	-78370.5	2522.28
LM-3	-83155.4	4069.01	-78367.0	2521.39
LM-4	-83440.2	4186.50	-78408.6	2531.90
LM-5	-82942.2	3983.22	-78141.4	2465.18

4.2.3. Rescoring

In the recognition experiments, the output lattice of the recognizer has been rescored using the same language models that are not pruned. The perplexity values of the language models which have been given in Table 4.7 are mainly lower in the unpruned language models that are built for rescoring process. New perplexity values is shown in Table 4.8 below.

Table 4.8. Total log-probabilities and normalized perplexities of the unpruned language models

	Based on morphs		Based on stem-endings	
	Log-probability	Perplexity	Log-probability	Perplexity
LM-1	-80282.0	3053.26	-74352.9	1688.12
LM-2	-79531.8	2832.7	-74205.9	1663.50
LM-3	-78968.7	2677.68	-74131.7	1651.21
LM-4	-79153.3	2727.54	-74190.1	1660.87
LM-5	-78990.9	2683.63	-74153.5	1654.81

We've also built class-based n-grams using two different clustering algorithms based on morph bigrams [5] and minimum edit distance [1]. Although we couldn't

perform speech recognition experiments using these models due to computation limitations, you can see the perplexity values in Table 4.9, which are very bad compared to the other language models.

Table 4.9. Perplexities of the class-based language models

	Total log-probability	Normalized perplexity
Based on morph bigrams	-96659.5	-97642.7
Based on minimum edit distance	15691.16	17311.41

4.2.4. Recognition Results

Recognition results in terms of Word Error Rates (WER) obtained by the experiments, that the morph recognition units are used while building the language models, are given in Table 4.10.

Table 4.10. Recognition results obtained by the morph-models

Count of less frequent word	WER	rescored WER
5	27.88	25.51
10	28.06	25.90
20	28.02	26.13
50	28.07	25.80
100	28.14	25.67

Results of the experiments that were performed using morph-stem-ending based language models are given in Table 4.11.

Table 4.11. Recognition results obtained by the morph-stem-ending-models

Count of less frequent word	WER	rescored WER
5	26.86	25.33
10	26.97	25.32
20	26.92	25.24
50	27.15	25.42
100	27.04	25.31

While in all of the experiments of which the results are given above, a vocabulary

size of 200,000 was used for building the language models; another experiment was performed with a vocabulary size of 50,000 for the data set which is segmented by the 3rd scheme and the language model is built upon morph-stem-ending units (taking only the words that are encountered more than 20 in the learning phase of the morph segmentation) to compare. WER of the experiment has been obtained as 27.28 and the rescored WER has been obtained as 25.64, which are slightly worse than the previous experiments.

After obtaining the best results with the stem-ending based models, we've performed a final experiment using a supervised stemmer called snowball [27]. Stemmers are the morphological analysers that only tries to extract the stems from words. Resulting WER was 27.78 and the rescored WER was 25.84, which are again comparable to the preceding results.

5. CONCLUSIONS

In this thesis, we have investigated the effect of automatically derived subword units to the performance of the LVCSR system. We've also made experiments using the subword units that are derived in a supervised manner and compared the results of two on the same experiment set.

We've mainly used an unsupervised morpheme analyser called Morfessor in our experiments with different parameters and segmentation approaches. We've also manipulate the Morfessor to model Turkish better by not allowing the subwords to be tagged as prefixes; however this approach has not improved the speech recognition performance.

Thinking that the larger the corpus we use, the better results we get; we've used a large corpus in the segmentation process. This approach had its own drawbacks like the high noise in the corpus. In order to cope with that problem we've introduced word token thresholds to the corpus before the learning process begins. Looking at the F-measure, we've obtained the closest segmentation to the actual one with the word token threshold of 10, i.e., eliminating the words that are encountered less than 10 from the corpus. But the best recognition result has been obtained from the experiment that has been performed using the language model which is resulted from the segmentation experiment with the word token threshold of 20.

We've also used the stem-ending units as the basic recognition units of language models and the best results have been obtained from those experiments. After realizing that fact, we've performed another experiment using a supervised segmenter called snowball, but the result was not satisfactory.

We've also examined class-based n-gram models using the morphs that are also used in the experiment which gave the best result in the recognition experiments. We've automatically clustered the corpus using two different algorithms and built class-based

n-grams upon these classes. We couldn't perform speech recognition experiments using these language models due to computation limitations. Nevertheless we've evaluated these language models using the perplexity, but the result was so bad that is not even comparable with the others.

The best results we've obtained are given with compared to the results that were obtained using word based language models on the same experiment set are given in Table 5.1.

Table 5.1. Summary of the best recognition results

	WER
Morph based with the word token limit of 5	25.51
Stem-ending based with the word token limit of 20	25.24
Word based	26.90
Supervised stem-ending based	25.42

As the Table 5.1 shows, unsupervised morpheme discovery based stem-ending model had performed slightly better. The evaluation of the speech recognition experiments was also presented in [28] with comparing to other language modeling approaches. Results of the significance tests measured by the NIST MAPSSWE significance test comparing the different approaches is given in Table 5.2.

Table 5.2. Summary of the best recognition results

		1	2	3	4
Morph based with the word token limit of 5	1	.	> 0.05	> 0.05	< 0.01
Stemending based with the word token limit of 20	2		.	=	< 0.01
Supervised stem-ending based	3			.	< 0.01
Word based	4				.

In the table, the operator (*) at location (a,b) shows the relation between a and b as "a * b". For example "< 0.001" at (1,4) states that morph based model perform significantly better than the word based model at $p < 0.001$.

Class-based language models are yet an open research area. Although our results

of the class-based n-grams were very bad in perplexity, better results could be obtained using more suitable clustering algorithms. Also the recognition experiments using these models can be performed using more memory-efficient algorithms.

REFERENCES

1. Jurafsky, D., J. H. Martin, *Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition.* Prentice Hall, 2000.

2. Rabiner, L. R., "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition", *Proceedings of the IEEE*, Vol. 77, pp. 257-286, February 1989.

3. Rosenfeld, R., "Two Decades of Statistical Language Modeling: Where Do We Go From Here?", *Proceedings of the IEEE*, Vol. 88, pp. 1270-1278, August 2000.

4. Price, P. J., "Evaluation of Spoken Language Systems: The ATIS Domain", *Proceedings of the DARPA Speech and Natural Language Workshop*, pp. 127-129, June 1990.

5. Brown, P. F., V. J. D. Pietra, P. V. deSouza, J. C. Lai, R. L. Mercer, "Class-based N-gram Models of Natural Language", *Computational Linguistics*, 18(4):467-479, December 1992.

6. Kneser, R., H. Ney, "Improved Clustering Techniques for Class-based Statistical Language Modeling", *Proceeidngs of the European Conference on Speech Communication and Technology (Eurospeech)*, 1993.

7. Şimşek, R., *Örneklerle Türkçe Söz Dizimi (Turkish Syntax with Examples).* Kuzey Matbaacılık, 1987.

8. Oflazer, K., E. Göçmen, C. Bozsahin, *An Outline of Turkish Morphology.* October, 1994.

9. Güngör T., Sak H., Saraçlar M., "Turkish Language Resources: Morphological Parser, Morphological Disambiguator and Web Corpus", *Language Resources and*

Evaluation, 2008.

10. Koskenniemi, K., "Two-level Morphology: A General Computational Model of Word-form Recognition and Production", *Tech. Rep. Publication No. 11, Department of General Linguistics, University of Helsinki*, 1983.

11. Creutz, M., K. Lagus, "Unsupervised Discovery of Morphemes", *Proceedings of the 6th Meeting of the ACL Special Interest Group in Computational Phonology in cooperation with the ACL Special Interest Group in Natural Language Learning*, pp. 21-30, Philadelphia, July 2002.

12. Harris, Z., "From Phoneme to Morpheme", *Language*, 31:190-222, 1955. Reprinted in *Papers in Structural and Transformational Linguistics*, Reidel Publishing Company, Dordrecht, Holland, 1970.

13. Hafer, M. A., S. F. Weiss, "Word Segmentation with Letter Successor Varieties", *Information Storage and Retrieval*, 10:371-385, 1974.

14. Dejean, H.,"Morphemes as Necessary Concept for Structures Discovery from Untagged Corpora.", *Workshop on Paradigms and Grounding in Natural Language Learning*, pp. 295-299, Adelaide, 1998.

15. Elman, J. L., "Finding Structure in Time", *Cognitive Science*, 14:179-211, 1990.

16. de Marcken, C. G., *Unsupervised Language Acquisition*, PhD Thesis, MIT, 1996.

17. Goldsmith, J., "Unsupervised Learning of the Morphology of a Natural Language", *Computational Linguistics*, 27(2):153-198, 2001.

18. Creutz, M., *Induction of the Morphology of Natural Language: Unsupervised Morpheme Segmentation with Application to Automatic Speech Recognition*, PhD Thesis, Helsinki University of Technology, Finland, 2006.

19. Hirsimäki, T., M. Creutz, V. Siivola, M. Kurimo, S. Virpioja, J. Pylkkönen, "Un-

limited Vocabulary Speech Recognition with Morph Language Models Applied to Finnish", *Computer Speech and Language*, 2006.

20. Creutz, M., "Unsupervised Segmentation of Words Using Prior Distributions of Morph Length and Frequency", *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pp. 280-287, Sapporo, Japan, July 2003.

21. Creutz, M., K. Lagus, "Induction of a Simple Morphology for Highly-Inflecting Languages", *Proceedings of the 7th Meeting of the ACL Special Interest Group in Computational Phonology: Workshop on Current Themes in Computational Phonology and Morphology, held in Conjunction with the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pp. 43-51, Barcelona, Spain, July 2004.

22. Creutz, M., K. Lagus, "Inducing the Morphological Lexicon of a Natural Language from Unannotated Text", *Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR05)*, pp. 106-113, Espoo, Finland, June 2005.

23. Creutz, M., K. Lagus, "Unsupervised Models for Morpheme Segmentation and Morphology Learning", *ACM Transactions on Speech and Language Processing*, 2006.

24. Arısoy E., H. Sak, M. Saraçlar, "Language Modeling for Automatic Turkish Broadcast News Transcription", *International Conferance on Spoken Language Processing - Interspeech*, 2007.

25. Arısoy E., M. Saraçlar, "Speech Recognition for Turkish Broadcast News", *SIU, IEEE 15th Signal Processing and Communication Applications Conference*, 2007.

26. Kneissler, J., D. Klakow, "Speech Recognition for Huge Vocabularies by Using Optimized Sub-word Units", *Proceedings of the 7th European Conference on Speech*

Communication and Technology, EUROSPEECH 2001, pp. 69-73, Aalborg, Denmark, 2001.

27. Eryiğit G., E. Adalı, "An Affix Stripping Morphological Analyzer for Turkish", *Proceedings of the IAESTED International Conference Artificial Intelligence and Applications*, Innsbruck, Austria, February 2004.

28. Aksungurlu T., S. Parlak, H. Sak, M. Saraçlar, "Comparison of Language Modeling Approaches for Turkish Broadcast News", *SIU, IEEE 16th Signal Processing and Communication Applications Conference*, 2008.