ISTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

TURKISH MORPHOLOGICAL DISAMBIGUATION USING MULTIPLE CONDITIONAL RANDOM FIELDS

M.Sc. THESIS

Razieh EHSANI

Department of Computer Engineering

Computer Engineering Programme

Thesis Supervisor: Prof. Dr. Eşref Adalı

JANUARY 2013

ISTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

TURKISH MORPHOLOGICAL DISAMBIGUATION USING MULTIPLE CONDITIONAL RANDOM FIELDS

M.Sc. THESIS

Razieh EHSANI
(504091523)

Department of Computer Engineering

Computer Engineering Programme

Thesis Supervisor: Prof. Dr. Eşref Adalı

JANUARY 2013

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ÇOKLU KOŞULLU RASSAL ALANLAR KULLANARAK TÜRKÇE İÇİN BİÇİMBİLİMSEL BELİRSİZLİK GİDERME

YÜKSEK LİSANS TEZİ

Razieh EHSANI
(504091523)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Eşref Adalı

OCAK 2013

Razieh EHSANI, a M.Sc. student of ITU Institute of Science and Technology 504091523 successfully defended the thesis entitled **"TURKISH MORPHOLOGICAL DISAMBIGUATION USING MULTIPLE CONDITIONAL RANDOM FIELDS"**, which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Prof. Dr. Eşref Adalı** Istanbul Technical University	.....
Co-advisor :	**Asst. Prof. Dr. Gülşen Eryiğit** Istanbul Technical University	.....
Jury Members :	**Asst. Prof. Dr. Ahmet Cüneyd Tantuğ** Istanbul Technical University	.....
	Prof. Dr. A. Coşkun Sönmez Yıldız Technical University	.....
	Assoc. Prof. Dr. Deniz Yüret Koç Univerity	.....

Date of Submission : **17 December 2012**
Date of Defense : **17 January 2013**

FOREWORD

The struggle itself towards the heights is enough to fill a man's heart. One must imagine Sisyphus happy.

"Albert Camus"

January 2013

Razieh EHSANI
Computer Engineer

TABLE OF CONTENTS

Page

ABBREVIATIONS

NLP	**:** Natural Language Processing
CRF	**:** Conditional Random Field
HMM	**:** Hidden Markov Model
MEMM	**:** Maximum Entropy Markov Model
MD	**:** Morphological Disambiguation
POS	**:** Part of Speech
DML	**:** Distinguishing Marker List

LIST OF TABLES

LIST OF FIGURES

Page

TURKISH MORPHOLOGICAL DISAMBIGUATION USING MULTIPLE CONDITIONAL RANDOM FIELDS

SUMMARY

Natural language processing is a branch of computer science that tackles different text processing tasks such as machine translation or query answering systems or many others. Turkish is an agglutinative language that has complex morphology. This property yields to the difficult of using the tools basically developped for English. We deal with Turkish morphology in this thesis. One word in Turkish, related to context, may have different morphologic properties, for example one word in one context is noun while in another context is verb. We tackle this problem to disambiguate morphological ambiguity. Our preferred approach for this goal is based on Machine Learning (ML). ML approaches are successful in many fields in computer science, including NLP. In this regard, we use a popular statistical approach to solve this problem. The Conditional Random Fields (CRFs) are a class of statistical modeling methods widely used in several NLP tasks. Compared with the other statistical approaches such as Hidden Markov Models and Maximum Entropy Markov Models, we use CRFs because they are more compatible with the nature of the morphological disambiguation problem. Also, CRFs are robust to over-fitting problem, since the number of parameters of the model is relatively less. CRFs can solve Label Bias problem because the normalization is performed at the sentence level. Furthermore, the likelihood function is convex, which means the global optimum can always be found using gradient based methods. Consequently, CRF is a successful method for sequence classification. Using CRFs, one can explicitly specify desired conditional dependencies. We define the linguistic features for our modeling. These features will be defined as edge features and node features on the rest of thesis. We used minimum Redundancy Maximum Relevance (mRMR) algorithm for choosing the relevant features. This algorithm shows successful effect in Part of Speech tagging (POS tagging) problem. Before dealing with the main problem of morphological disambiguation, we also tackle the POS tagging problem in this thesis. POS tagging problem is a part of morphological disambiguation problem, that deals with only the main tags whereas in full morphological analysis, the stem, the structure and inflectional features are also determined together with the main tags. Regarded to POS tagging problem, we selected features and used them in morphological disambiguation problem. The main reason of this is to replace the problem with a simpler one which has shorter training time.

During this work, we were faced with different problems such as inference time, modeling Turkish morphological structure and selecting good features. To use the CRF method, we employed the well-known MALLET library. MALLET library implements the Viterbi algorithm for inference. This “pure” Viterbi method is

unsuitable for our purposes, for this reason we use our implementation of constrained Viterbi instead.

There are 116 morphological tags in Turkish, if we want to model all of them as one group of tags, we face with a significant problem where the training duration may be up to 2 weeks. We solve this problem by dividing the tags into 9 separate groups. All of this categories have special properties. For example, the main part of speech tags category contains main part of speech tags, or the tense category contains tags related to time and tenses. With this modeling, training duration reduces to a day if we do not use minor part of speech tags. For the feature selection, we use tools such as mRMR and MALLET feature induction, although these methods show insufficient performances which we will discuss in the rest of the thesis. .

ÇOKLU KOŞULLU RASSAL ALANLAR KULLANARAK TÜRKÇE İÇİN BİÇİMBİLİMSEL BELİRSİZLİK GİDERME

ÖZET

Bilgisayarlı dil bilimi, bir dilin yapısal veya istatistiksel özelliklerini inceleyerek o dile ait verileri işleyip belli başlı sorunlara çözüm aramak için birçok konuyu inceleyen bir bilim dalıdır. Bu konular arasında başta, bilgisayar bilimleri, dil bilimleri, bilişsel bilimler ve felsefe gelmektedir. Bilgisayarlı dil bilimlerinde amaç, dilin yapısal özelliklerine ilişkin kuramsal çıkarımlar yapmakla birlikte dili modellemek ve işlemek suretiyle uygulamada bazı faydalı çözümler üretmektir. Bu konudaki öncü çalışmalar 1950'li yıllarda Bilgisayarlı Çeviri alanında görülmüştür. Bu arastirmalarin ortaya cikis sebebi daha cok Sovyetler Birligi'nde yayinlanan makaleleri cevirmek sureti ile endustriyel casusluk yapmaktir. Ancak alanin gelismesi sonucunda farkli uygulama alanlari da belirmistir. Bugun makine cevirme alaninda yapilan arastirmalarin urunu olarak "google translate" gibi basarili makine cevirme uygulamalari mevcuttur.

Doğal dil işleme, bilgisayarlı dil bilimi için önemli alanlarından biridir. Doğal dil işlemenin amacı, dili pratik bir amaca hizmet etmek için modellemektir. Kuramsal hesaplamalı dil, diğer bilimsel çalışmalardan farklı olarak doğal dil işlemede dilin modellenmesindeki karmaşıklık, hizmet edecek amaca uygun olarak değişebilir. Dolayısıyla burada amaç dili mümkün olduğunca iyi modellemek değil istenen amacı mümkün olduğunca başarılı bir şekilde gerçekleştirmektir. Bilgisayarlı Çeviri, biçimbilimsel inceleme, biçimbilimsel belirsizlik giderme, anlamsal belirsizlik giderme, bilgi çıkarımı gibi konular doğal dil işlemenin önemli konuları arasındadır. Genelde iki temel yaklaşım olduğu gözlenebilir. Bunlardan ilki, dilin belirli önemli yapısal özellikleri öne çıkarılarak elle belirlenen veya otomatik çıkarılan kurallar yoluyla, istenen amaç gerçekleştirilir. Bu tip calismalar daha cok Noam Chomsky ekolu ile ozdeslestirilir. ozellikle 1960-80'li yillarda populer olan bu yaklasim, daha sonra faydaci bazi kaygilar nedeni ile arka plana itilmistir. Bugun daha populer olan diğer bir yaklaşım ise dili çeşitli gelişmiş istatistik ve makine öğrenmesi yöntemleri ile modellemektir. Bu alandaki calismalarin bir kismi, dili olasiliksal modellemek sureti ile cikarimlar yapmaya dayali iken, baska bir yaklasim "kapali kutu" algoritmalar kullanarak dogrudan istenilen davranisin bilgisayar tarafindan ogrenilmesidir. Bizim çalışmamız bu ikinci yaklaşımı benimsemektedir.

Özellikle karmaşık biçimbilimsel özellikler gösteren dillerde (Türkçe, Fince, Çekçe) biçimbilimsel çözümleme ve belirsizlik giderme konuları önemlidir. Biçimbilimsel belirsizlik giderici, Türkçede, diğer doğal dil işleme konularında bir önişleme olarak ele alınmaktadır. Türkçedeki biçimbilimsel belirsizlik, Türkçenin zengin biçimbilimsel özelliğinden kaynaklanmaktadır. Türkçe bir kelime teorik olarak sonsuz sayıda ek alabilmekte ve Türkçenin aldığı her ek ile kelimenin biçimbilimsel özelliği değişebilmektedir. Bu zenginlik, kelimelerin cümle içinde aldıkları konuma

göre daha çok belirsizliğe neden olmaktadır. Bazı Türkçe kelimelerin 20'nin üzerinde biçimbilimsel çözümlemelere sahip olduklarını görebiliyoruz. Biçimbilimsel çözümleme, Türkçede 116 etiketten oluşmaktadır. Her kelimenin cümledeki konumuna bakılmaksızın bu etiketlerden oluşan bir ek, biçimbilimsel çözümlemede ele alınmaktadır. Bu etiketlerden 12'si Part of Speech olarak, kelimenin sıfat, isim, fiil ve zarf olması gerektiğini belirtir. Biçimbilimsel belirsizlik giderici, biçimbilimsel çözümleyicinin belirlediği olası biçimsel çözümlemelerin arasından doğru olanı seçme problemidir. Bu problemi çözmek için, kelimenin içinde bulunduğu bağlam değerlendirilir. Ana etiket belirsizliği giderme ise kelimenin cümlede aldığı konuma göre ana etiketler kümesinden alabileceği etiketi belirleme yöntemidir. Bu sorun İngilizce gibi dillerde çok karmaşık bir sorun oluşturmaz. Fakat, Türkçede ise aşılması güç olan sorunlara neden olmaktadır. Çalışmamızın amacı, Türkçenin hem ana etiket belirsizliği hem de biçimbilimsel belirsizliğini gidermektir. Bu sorunu daha önce yapılan çalışmalardan farklı olarak istatistiksel makine öğrenmesi yöntemi ile ele almaktayız. Son zamanların doğal dil işleme çalışmalarında yer alan koşullu rassal alanlar yöntemi bu çalışmada kullanılmıştır. Bu yontem Turkce icin daha once de uygulanmaya calisilmis da olsa, bu denemeler istenen basarimi saglayamamistir.

Koşullu rassal alanlar, bir koşullu olasılık dağılımıdır. Koşullu rassal alanlar yaklaşımında biçimbilimsel çözümlemeler, kelimelere koşullu olarak bir olasılık atamaya çalışmaktadır. Biçimbilimsel çözümlemelerin arasındaki herhangi bir biçimbilimsel veya istatistiksel ilişki, koşullu rassal alanlar için bir özellik olarak kullanılmaktadır. Ayrıca biçimbilimsel çözümlemelerin ve kelimelerin arasındaki istatistiksel ve biçimbilimsel özellikleri de kullanılmaktadır. Bu özelliklerin ağırlıkları, öğrenme verisinden öğrenilmektedir. Bu çalışmanın temel konularından biri bu özelliklerin tanımı ve yararlı özelliklerin seçilmesidir. Çünkü seçilen doğru özellikler başarıyı daha da yükseltmektedir. Koşullu rassal alanlar, parametre öğrenmede L-BFGS algoritmasını, çıkarım kısmında ise Viterbi algoritmasını kullanmaktadır. Kosullu rassal alanlarda ogrenme problemi, ussel dagilim ailesinde en buyuk olabilirlikli (maximum likelihood) parametre ogrenme yonteminin bir ozel durumudur ve bu genel problemin bir ozelligi olan icbukeyligi tasir. Bu ozellik, parametre eniyilemesinin biricik cozumu oldugunu ve bu cozume Newton-tipi yontemlerle ulasilabilecegini guvence altina alir. L-BFGS algoritmasi, dusuk bellek kullanan ve genellikle hizli yakinsama yaptigi gozlemlenen sozde-Newton yontemlerinden birisidir.

Bu çalışmada zincir koşullu rassal alanlar kullanılmıştır. Zincir koşullu rassal alanlar bir graftaki komşulukları göz önüne almaktadır. Öğrenme ve deneme amacıyla MALLET aracı kullanılmıştır. Bu araç ayrıca koşullu rassal alanların doğası gereği yavaş ve zaman alıcı bir araçtır. Bu çalışmada ayrıca daha başarılı ve daha hızlı sonuca varmak için çeşitli yöntemler geliştirilmiştir. Bu yöntemler, hem ana etiket atama probleminde hem biçimbilimsel belirsizlik gidericide koşullu rassal alanların cümle bazında optimizasyon yapmasını mümkün kılmıştır ve bu sebepten dolayı başarıyı da ayrıca yükseltmiştir. Ana etiket atama probleminde bir tek koşullu rassal alan kullanılırken biçimbilimsel belirsizlik gidericide birçok koşullu rassal alan kullanılmıştır. Biçimbilimsel belirsizlik giderici, 116 etiketi 9 ayrı kümede toplamıştır. Bu 9 küme, Türkçenin biçimbilimsel özelliğine göre düzenlenmiştir. Biçimbilimsel

belirsizlik gidericide, bu 9 küme için koşullu rassal alanlar ayrı ayrı eğitilmiştir. Bu eğitilmiş koşullu rassal alanlar, daha sonra birleştirilerek problem çözümlenmektedir.

Daha önceki 2 çalışmaya bakılırsa, birincisi kural tabanlı bir çalışmadır, bu çalışmanın dezavantaji ise kelime bazında eniyileme yapmasıdır. İkinci çalışma ise Perceptron algoritmasından yararlanmıştırr, ilkine göre daha yüksek başarı elde edilmiştir. Burada Perceptron algoritması HHM kullanan bir çalışmanın çıktıları üzerine kullanmak sureti ile hızlı ve başarılı bir yöntem gerçeklenmiştir. Bizim tekrar bu konuyu ele almamızın amacı ise, istatistiksel makine öğrenmesi kullanarak bu sorunu çözmektir. Ayrıca başarıyı yükseltmek ve çözümü pratik bir hala getirmek için farklı farklı yöntemler kullandık. Bu arada ne kadar çalışmanın dilden bağımsız olması doğrultusunda çalışsakta, Türkçenin tüm özelliklerini birer birer incelemiş ve ele almış olduk. Bu çalışmada ne kadar hazır kütüphaneler kullanılsada bir çok kütüphaneyi değiştirmek ve baştan yazmak zorunda kaldık. Örnek olarak MALLET'te Viterbi algoritmasını tamamen değiştirip ve onun yerine Constrained Viterbi algoritmasını baştan yazdık. Daha önce başka diller için yapılan benzeri çalışmalarda koşullu rassal alanların kullanılmasını pratik bulamamışlardır. Özellikle Çekce için bunun pratik bir çözüm olduğunu gösterememişlerdir. Biz bu zorluğu aşmış durumdayız. Çalışmamız için sorunu kaç farklı probleme ayırıp ve öyle ele almak ise bizim bu sorunu pratik bir çözüm önermemiz için avantajlı kılmıştır. Bu çalışmanın tekrarlanabilmesi ve geliştirilebilmesi için özen gösterdik. Koşullu rassal alanları kullanmamızın asıl amacı istatistiksel makine öğrenmesinde, diğer yöntemlere göre avantajlarının olmasıydı. Örneğin koşullu rassal alanlar gizli markov modellerine ve en buyuk entropi markov modellerine göre avantajlara sahip. Bu avantajlardan koşullu rassal alanların gizli markov modellerine göre çok daha az parametre ile uğraşmasıdır. En buyuk entropi markov modellere göre ise koşullu rassal alanlarda label bais sorunun olmamasıdır. Bu sorun ağırlıkların olasılık cinsinden olmasından kaynaklanıyor oysa koşullu rassal alanlarda ağırlıklar olasılık cinsinden değildir.

Ana etiket atama probleminde yüzde 98.60 oranında bir başarı elde edilmiştir. Biçimbilimsel belirsizlik gidericide ise bu başarı yüzde 95.57'dir. Bu degerler literaturdeki en iyi basarim oranlarina oldukca yakindir.

1. INTRODUCTION

Communication is a crucial part of any social organization and as the technology advances, the benefits are also noticed in this area. The benefits include the ability to communicate further and faster than before and with more people simultaneously. But the advanced technology does not serve only as a more efficient carrier of human communication signals but also as effective processors of these signals. It is the task of the field of Natural Language Processing (NLP) to derive important characteristics of a communication signal and consequently process it.

The existence of different models of communications, such as different languages, creates various challenges. Sometimes, it might not be possible to derive a method that works best for any language, in such problems domain/language dependent studies are regarded. Turkish language has a very rich morphological structure and as an agglutinative language, shows very different characteristics compared to English. One example is the property that the words in Turkish can, theoretically, take infinite number of suffixes and a suffix may change the semantic or syntactic properties of a given word drastically.

Analyzing morphological properties of a given sentence is a crucial task for many subsequent processing, such as parsing and word-sense disambiguation and many other supervised methods in NLP. However, when the analysis is done at a word level, one usually gets multiple possible analysis results to choose from. A full analysis of a word contains morphological properties such as Part-of-Speech (POS) tag, tense, plurality, etc. This problem of ambiguity can only be solved using contextual information in terms of the sentence involved or maybe even a larger unit.

According to our calculation, the ratio of the ambiguous words to all words is 50% in the corpus of Turkish sentences which we used in our studies. That means a morphological analyzer will fail to unambiguously identify the correct analysis using the word features alone. Moreover, some of the words in the corpus can have up to 23

different morphological analysis result. This complex structure of Turkish language in terms of its morphological properties makes the morphological disambiguation problem a highly difficult one.

The morphological disambiguation problem for morphologically rich languages differs significantly from the well-known POS tagging problem. It is rather an automatic selection process from multiple legal analysis results of a given word than the assignment of a POS tag from a predetermined tag set. The possible morphological analysis results of a word (generally produced by a morphological analyzer) in morphologically complex languages are very complex when compared to morphologically simple ones: They consist of the lemma, the main POS tags and the tags related to the inflectional and derivational affixes. The number of the set of possible morphological analysis results may sometimes be infinite for some languages such as Turkish.

In this study, we firstly focus on the determination of the main POS tags (which will be referred as "POS tagging" from now on) and in the next step the full disambiguation task by using conditional random fields. There are few methods for Turkish which directly tackle the POS tagging problem. Instead many methods perform a full morphological disambiguation and the POS tags are obtained from the correct parses. In this work, we take a different approach and propose a model which directly tackles the POS tagging problem by using conditional random fields. In addition to this, by employing the opinion pooling method discussed in [1], we generalize this approach to full morphological disambiguation.

To give a sense of the problem at hand and the general morphological disambiguation, we have measured the ambiguity corresponding to the POS tagging and Morphological Disambiguation problems. About 27% of the words in our corpus are ambiguous in terms of its POS tag and random guessing has an expected accuracy of 85%, on the other hand the ambiguity in terms of morphological disambiguation is about 50%. The proposed approach in this thesis improves the accuracy of POS tag to around 98.60%, and accuracy of full morphological disambiguation around 95.57%.

Our approach is based on the well-known methodology of Conditional Random Fields, which is also applied to other languages with varying success. POS tagging problem was successfully tackled in languages with relatively simpler morphological properties (such as English) [1, 2, 3, 4]. On the other hand, other languages proved to be more problematic with lower tagging performance, [5, 6, 7] with accuracies ranging from 85% to 95%. Smith et. al. [1] discusses the high computational burden of CRFs in both training and inference steps and argues that this is a major obstacle in its practical usage. In this work, we also discuss performance related issues and propose different approaches to lower the computational burden in inference step. The best approach among these is very close to the state of the art [8] in performance, while being competitive in computational complexity. We also discuss the problem of feature selection in order to reduce training times and improve generalization capability. We employ the well-known mRMR [9] method to this end. These efficiency improvements are important steps toward making CRFs more practical tools in NLP. The improvements result in the reduction of training time, and also a slight increase in accuracy.

2. BACKGROUND ON MORPHOLOGICAL DISAMBIGUATION

In this chapter, we shall introduce the Turkish morphological properties. Understanding these properties are very important for modeling Turkish language. We will also use this information in the feature selection3.1.

2.1 Morphological Properties of Turkish Sentences

Turkish is an agglutinative language which has a complex morphological structure. This property of the Turkish language leads to vast amounts of different surface structures found in texts. In a corpus of ten million words, the number of distinct words exceeds four hundred thousand [10]. There are several suffixes, which may change the POS tags of the words from noun to verb or verb to adverb, etc. Thus, it is much harder to determine the final POS tag of a word using the root such as in English. Because of this, we cannot resort to lexicons of words (roots) as in many studies on English. We must use the morphological analysis of the words to determine the tags. The context dependency of tags of words must also be taken into account.

There are several tags which determine respective properties of the associated words. These tags contain syntactic and semantic information and are called morphosyntactic or morphosemantic respectively. We use the same representation for the tags as [11]. Any words in Turkish can be represented by the chain of these tags. We call these chains of tags for words morphological analysis results of these words.

Turkish morphological analysis considers 116 different tags. To better model these tags and circumvent the data sparseness problems, we have partitioned these into 9 disjoint groups, called slots. The slots are determined such that the semantic relation among the tags in a slot is maximum, while it is minimum for tags across slots. Also a word cannot accept more than one tag from a single slot. Essentially transforming the problem into a multiple class classification problem. Such a construction of the problem, with this particular slot partitioning, is one of the contributions of the thesis.

The main properties of the words are expressed in the main POS category and the other slots serve to fill in the details such as plurality, tense, etc. In this thesis, one of the problems that we are concerned with is the correct disambiguation of the main POS tags. So we are interested in identifying the value of a single slot. We also deal with the morphological disambiguation problem, for which we also model other slots and devise a mechanism to combine these slots.

Many words in Turkish texts have more than one morphological analysis. Sometimes the number of possible morphological analysis results reach 23. Because of derivative and inflective properties at the Turkish language, in theory, one word can use an infinite number of suffixes. Due to this, we are faced with an immense vocabulary in Turkish. The large vocabulary size causes data sparseness problem. Some of these suffixes change the word meanings. In this case, these changes are expressed with inflectional groups (IGs) that are separated by ˆ*DB* sign, where ˆ*DB*'s mean derivation boundary (root+IG1+ ˆDB+IG2+ ˆDB+...+ ˆDB+IGn). One Turkish word can have many IGs in its morphological analysis results. These IGs and the related tags can also be represented as tags. The standard morphological tags, also used in this work, are shown in Table 2.1. The example below shows the morphological analysis results for the word "alındı" produced by a Turkish two-level morphological analyzer [12].

1. al+VerbˆDB+Verb+Pass+Pos+Past+A3sg (It was taken)
2. al+AdjˆDB+Noun+Zero+A3sg+P2sg+NomˆDB+Verb+Zero+Past+A3sg (It was your red)
3. al+AdjˆDB+Noun+Zero+A3sg+Pnon+GenˆDB+Verb+Zero+Past+A3sg (It was the one of the red)
4. alındı+Noun+A3sg+Pnon+Nom (receipt)
5. alın+Verb+Pos+Past+A3sg (resent)
6. alın+Noun+A3sg+Pnon+NomˆDB+Verb+Zero+Past+A3sg (It was the forhead)

We categorized all tags in morphological analysis into 9 independent groups. First group is the main part of speech group, that determines general morphological

Table 2.1: Morphological Tags

Slot Groups	Slot Values
Main POS	Adj, Adv, Conj, Det, Dup, Interj, Noun, Num, Postp, Pron, Punc, Verb
Minor POS	Able, Acquire, ActOf, Adamantly, AfterDoingSo, Agt, Almost, As, AsIf, AsLongAs, Become, ByDoingSo, Card, Caus, DemonsP, Dim, Distrib, EverSince, FeelLike, FitFor, FutPart, Hastily, InBetween, Inf, Inf1, Inf2, Inf3, JustLike, Ly, Ness, NotState, Ord, Pass, PastPart, PCAbl, PCAcc, PCDat, PCGen, PCIns, PCNom, Percent, PersP, PresPart, Prop, Quant, QuesP, Range, Ratio,Real, Recip, ReflexP, Rel, Related, Repeat, Since, SinceDoingSo, Start, Stay, Time, When, While, With, Without, Zero
Person Agreements	A1pl, A1sg, A2pl, A2sg, A3pl, A3sg
Possessive Agreements	P1pl, P1sg, P2pl, P2sg, P3pl, P3sg, Pnon
Case Markers	Abl, Acc, Dat, Equ, Gen, Ins, Loc, Nom
Polarity	Neg, Pos
Tense/Mood	Aor, Desr, Fut, Imp, Neces, Opt, Pres, Prog1, Prog2, Cop, Cond, Past, Narr
Compund Tense	Comp_Cond, Comp_Narr, Comp_Past
Cop	Cop

properties of words. In this group, each tag determined major POS of word as shown in Table 2.2

Table 2.2: Main Part of Speech

Morphological Tag	Description	Morphological Tag	Description
Noun	Noun	Adj	Adjective
Adv	Adverb	Cond	Condition
Det	Determiner	Dup	Duplicator
Interj	Interjection	Verb	Verb
Postp	Postpositive	Num	Number
Pron	Pronoun	Punc	Punctuation

Second group is the Minor Parts of Speech of a word, that is content 65 morphological tags, these tags determine the minor morphological properties such as semantic markers, causative markers, post-position. We listed these tags with their description in Table 2.3

Third group is the Number/Person Agreement that indicated in Table 2.4.

Fourth group is the Possessive Agreement is indicted in Table 2.5.

Table 2.3: Minor Part of Speech

Morphological Tag	Description	Morphological Tag	Description
Able	able to verb	Acquire	to acquire the noun in the stem
ActOf		Adamantly	
AfterDoingSo		Agt	
Almost	almost verbed but did not	As	
AsIf		AsLongAs	
Become	to become like the noun or adj in the stem	ByDoingSo	
Card	Cardinal	Caus	Causative
DemonsP	Demonstrative Pronoun	Dim	Diminutive
Distrib	Distribution	EverSince	have been verbing ever since
FeelLike		FitFor	
FutPart	Future Participle	hastily	verb hastily
InBetween		Inf	Infinitive
JustLike		Ly	slowly
Ness	as in Red vs Redness	NotState	
Ord	Ordinal	Pass	Passive
PastPart	Past Participle	PCAbl	
PCAcc		PCDat	
PCGen		PCIns	
PCNom		Percent	Percentage
PersP	Personal Pronoun	PresPart	Present Participle
Prop	Porper Noun	Quant	Quantifying
Ques	Question	Range	Range
Ration	Ratio	Real	
Recip	Peciprocal	ReflexP	Reflexsiv Pronoun
Rel		Related	
Repeat	verb repeatedly	Since	
SinceDoingSo		Start	start verbing immediately
Stay	stayed	Time	Time
When		While	
With		Without	
WithoutHavingDoneSo		Zero	

Table 2.4: Number/Person Agreement

Morphological Tag	Description	Morphological Tag	Description
A1sg	1.singular	A2sg	2.singular
A3sg	3.singular	A1pl	1.plural
A2pl	2.plural	A3pl	3.plural

Table 2.5: Possessive Agreement

Morphological Tag	Description	Morphological Tag	Description
P1sg	1.singular	P2sg	2.singular
P3sg	3.singular	P1pl	1.plural
P2pl	2.plural	P3pl	3.plural

Fifth group is the Case Marker group is indicated in Table 2.6.

Table 2.6: Case Marker

Morphological Tag	Description	Morphological Tag	Description
Nom	Nominative	Acc	Accusative
Dat	Dative	Abl	Ablative
Loc	Locative	Gen	Genitive
Ins	Instrumental	Equ	Equative

Sixth group is the verb Polarity markers shown in Table 2.8.

Seventh group is the "tense", the "aspect" and the "mood" group indicated in Table **??**.

Verbs may have one or two such markers, for each slot we model the morphological tags given the input sentence conditionally independent of each other.

2.1.1 Evaluating POS tagging performance of Hasim Sak's work [8]

To allow comparison, we have prepared the following statistics using Sak's POS tagger.

The distribution of errors per POS tags is given in Table 2.10

An example for P3sg's error is "(pantolonu : P3sg|A3sg) dizlerine dek ıslak".

The confusion matrix is given for correct value vs. predicted value in the following tables.

Table 2.7: Tense and Aspect and Mood

Morphological Tag	Description	Morphological Tag	Description
Pos	Positive	Neg	Negative

Table 2.8: Verb Marker

Morphological Tag	Description	Morphological Tag	Description
Past	Past tense	Narr	Narrative past
Fut	future	Aor	Aorist, present
Pres	Present	Desr	Desire, wish
Cond	Conditional	Neces	Necessitative
Opt	Optative, let	Imp	Imperative
Prog1	process	Prog2	state

An example for Adj's error: "Ercan Tezer, (iç| Adj|Noun) pazarda bu yıl güzelliğini bile fark(edemez| Adj|Verb) hale gelmiştim. Gönlüm sizin bu kadar (çok|Adj|Det) acı çekmenize razı değil."

An example for Adv's error: "Kaç gündür bu (böyle|Adv|Adj). Kumral saçları (hafifçe|Adv|Adj) karışmıştı. Bence yeterince değil, hiç (araştırmadan|Adv|Noun) haber.

An example for Pron's error : "Kaç gündür (bu|Pron|Det) böyle tabanı ne derse (o|Pron|Det) olacak. An example for Num's error: "İhracat bedeli (yüzseksen|Num|Noun) arttı."

Table 2.9: Statistical Analysis

All word count	45999
Correct word count	39048
Incorrect word count	6951
accuracy	84.89

Table 2.10: Error Analysis

POS tag	Count	Incorrect	Accuracy
P3sg	11361	868	7.64
Adj	9153	800	8.7
Adv	6126	566	9.2393
P3Pl	938	417	42.4211
Pron	2267	308	13.586
Num	707	226	31.98
Persp	1157	261	22.55
P2sg	329	168	51.06

Table 2.11: Error Analysis for Adj

POS tag	Incorrect POS	Mismatch count
Adj	Noun	501
Adj	Verb	139
Adj	Det	99
Adj	Adv	42

Table 2.12: Error Analysis for Adv

POS tag	Incorrect POS	Mismatch count
Adv	Adj	305
Adv	Noun	97
Adv	Det	68
Adv	Postp	57

Table 2.13: Error Analysis for Pron

POS tag	Incorrect POS	Mismatch count
Pron	Det	152
Pron	Adj	72
Pron	Noun	57
Pron	Adv	27

Table 2.14: Error Analysis for Num

POS tag	Incorrect POS	Mismatch count
Num	Noun	113
Num	Det	111

3. FEATURE SELECTION

One method to improve the performance of a machine learning method is to select a subset of informative features [13]. A good feature selection method can improve variance of the estimates without introducing a significant bias. The minimum Redundancy Maximum Relevance (mRMR [9]) method relies on the intuitive criteria for feature selection which states that the best feature set should give as much information regarding the class variable as possible while at the same time minimize inter-variable dependency as much as possible (avoiding redundancy). The two concepts, relevance and redundancy, can be naturally expressed using information theoretic concept of mutual information. However, real data observed in various problems are usually too sparse to correctly estimate the joint probability distribution and consequently the full mutual information function. The solution proposed in [9], employs two different measures for redundancy (*Red*) and relevance (*Rel*):

$$Red = 1/|S|^2 \sum_{F_i,F_j \in S} MI(F_i,F_j) \tag{3.1}$$

$$Rel = 1/|S| \sum_{F_i \in S} MI(F_i,R) \tag{3.2}$$

In the expressions above, S is the set of features of interest, $MI(.,.)$ is the mutual information function, R is the class variable and F_i is the random variable corresponding to the ith feature. Then the goal of mRMR is to select a feature set S that is as relevant ($\max(Rel)$) and as non redundant ($\min(Red)$) as possible. In the original work (mRMR) [9], two criteria to combine *Rel* and *Red* were proposed. In this work, the criterion of Mutual Information Difference ($MID = Rel - Red$) is used, because it is known to be more stable than the other proposed criterion ($MIQ = Rel/Red$) [14].

As a side note, we have also considered the "feature induction" in [4]. However, we have observed a significant drop in accuracy and therefore will not discuss this approach in this thesis.

3.1 Features

In a linear chain conditional random field, there are two types of features: edge features and node features. Edge features are functions of labels of consecutive words ($f_k(y_i, y_{i+1})$) and node features are functions of words in the sentence ($f_k(y_i, \mathbf{x})$, where $\mathbf{x}$ denotes words of the sentence). The probability of a sequence is determined by the feature values as well as the associated model parameters. Thus, determining good feature functions that describe the important characteristics of the words is crucial for a successful model. We employ several morphological/syntactical properties as features.

In our model, the feature functions f_k are determined using several tests such as capitalization, end of sentence, etc. Results of these tests together constitute the features vector $F = f_1, f_2, ..., f_k$ for a word.

To illustrate the two kinds of features, let's consider one feature for node and edge type features used in our model. The *Color* feature is an example for a node feature, it is a function that returns one if the word is among a set of words describing colors and zero otherwise. The indicator function $\Phi(y_i = Adj, y_{i+1} = Noun)$, which returns one if the expression is true and zero otherwise, is an example of an edge feature.

The edge functions in our proposed method consist of all possible slot value pairs. The node functions are given in Table 3.1. The features "Color Set Feature", "Digit Set Feature", "Pronoun Set Feature", "Transition Set Feature" and "Non-Restrictive Set Feature" indicate whether the word is a member of corresponding sets of special words. These sets correspond to specific linguistic classes in Turkish language. The "Noun Adj Feature" indicates whether the word has suffixes that are generally used to change a noun to an adjective. "Capital Feature" indicates whether the word starts with a capital letter. "Before amount feature" and "Before Ques Morpheme Feature" indicate whether the word is followed by a special word/class of words. As their names imply, "Beginning Sentence Feature" and "End Sentence Feature" indicate

Table 3.1: The features considered in this work

Feature Templates	**Number of Corresponding Features**
Capital_Feature	1
End_Sentence_Feature	1
Begining_Sentence_Feature	1
Color_Set_Feature	1
Equal_Slot_Feature	116
Digit_Set_Feature	1
Before_Mi_Feature	1
Pronoun_Set_Feature	1
Transition_Set_Feature	1
Nonrestrictive_Set_Feature	1
Before_Amount_Feature	1
Noun_Adj_Feature	1
X2Y_Before_slot	116
X2Y_After_slot	116
After_Capital_Feature	1
Proper_Feature	1
PostP_Feature	1
Apostrophe_Feature	1
Total	363

whether the word is at the beginning or the end of the sentence. Finally, “Equal Slot”, “X2Y Before” and “X2Y After” feature templates generate features based on whether respectively the word itself, the word before or after it has a particular slot value which is unambiguously known, i.e. These values are the same for all possible analysis results of the word. We have also considered looking into the previous two and the next two words, but it turned out to degrade the performance. These classes of features contain 363 feature functions. However, in application, some of these features were discarded using mRMR as explained in Section 3. Figure 3.1 shows a sample sentence and the corresponding features. In this Figure, we observe that the first word "Tebriz’in" gets the "Beginning" feature. Since the morphological analyzer states that the fact that this word is "A3sg", "Noun" and "Prop" unambiguously, i.e. these tags show up in all of the possible parses, we also have the "Equal Slot" generated features of "A3sg", "Noun" and "Prop". Finally, we see the feature "X2Y Before A3sg" which means the word after this one is unambiguously known to be "A3sg". We can confirm this by checking the next word "kışı" where we can see the feature "A3sg" as expected. The features for the other words can be understood similarly.

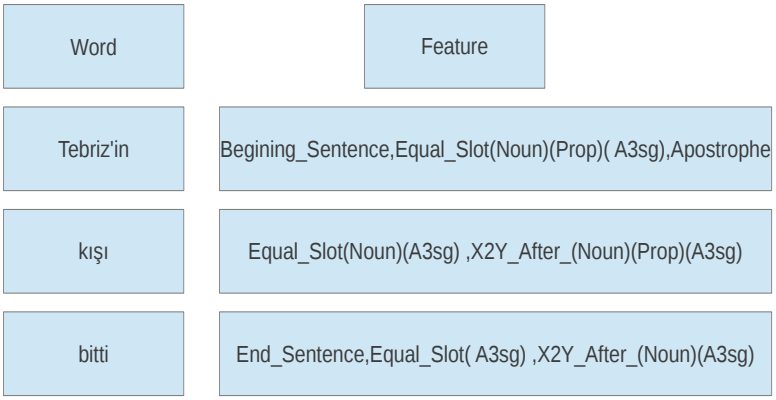

Figure 3.1: A sample sentence and the corresponding features

4. CONDITIONAL RANDOM FIELDS

A Conditional Random Field (CRF) is a conditional distribution $p(\mathbf{y}|\mathbf{x})$ in the form of a Gibbs distribution and with an associated graphical structure encoding conditional independence assumptions. Because the model is conditional, dependencies among the input variables x are not explicitly represented, enabling the use of rich and global features of the input (neighboring words, capitalization...). CRFs are undirected graphical models used to calculate conditional probability of realizations of random variables on designated output nodes given the values assigned to other designed input nodes. In the special case, where the output nodes of the graphical model are linked by edges in a linear chain, CRFs make a first-order Markov independence assumption. Thus, it can also be understood as a conditionally-trained finite state machine (FSM).

The distribution related to a given CRF is found using the normalized product of potential functions ($\Psi_C(\mathbf{y}_C)$) for each clique (C). The potential function itself can be, in principle, any non-negative function. Formally, the conditional probability $p(\mathbf{y}|\mathbf{x})$ can be expressed as

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \frac{1}{Z(\mathbf{x})}\Pi_C\Psi_C(\mathbf{y}_C,\mathbf{x}) \\ &= \frac{1}{Z(\mathbf{x})}exp(-\textstyle\sum_C H_C(\mathbf{y}_C,\mathbf{x})) \end{aligned} \tag{4.1}$$

On the above equations, $H_C(\mathbf{y}_C,\mathbf{x}) = \log(\Psi_C(\mathbf{y}_C,\mathbf{x}))$. A CRF can also be seen as a weighted finite state transducer [1]. For example, in Figures 4.1 and 4.2, we can see the equivalent expression of a linear chain (1st order) CRF and 2nd order CRF as finite state transducers. These figures clearly show the parameter explosion when the order is increased. Higher number of parameters denies us the possibility of accurate parameter explosion in finite data. Indeed, using CRFs with order greater than one, deteriorates the model performance. On the other hand, a CRF of order 0 discards all neighborhoods information, effectively eliminating the advantages of sequential modeling. Unlike MEMM (see [15]), the transition weights in CRF are non

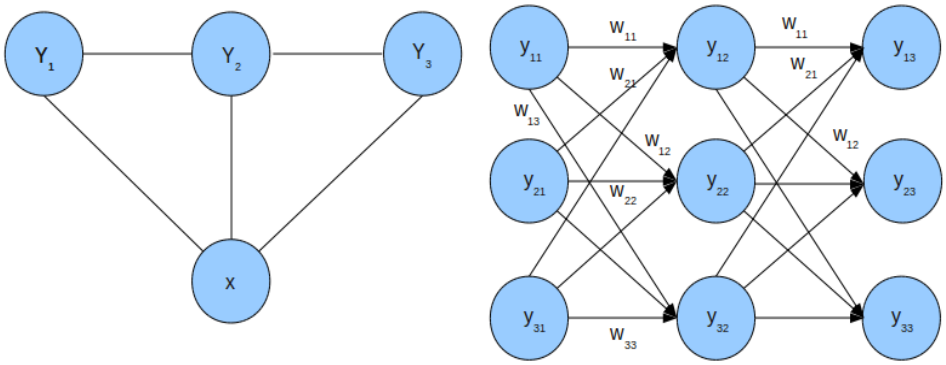

Figure 4.1: The equivalent expression of a linear chain CRF (on the left) as a FST (on the right)

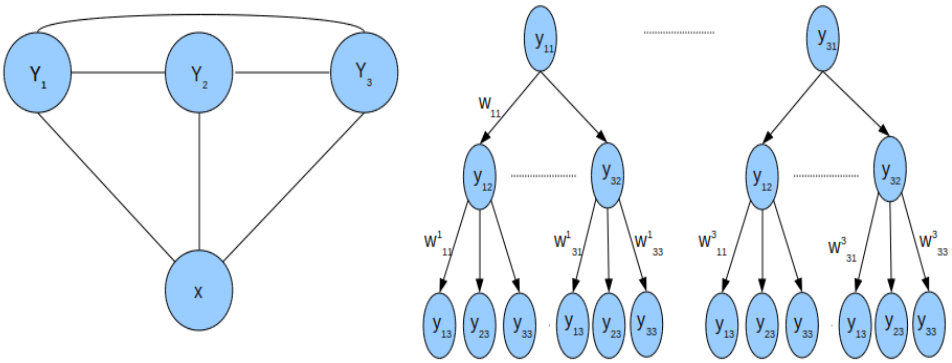

Figure 4.2: The equivalent expression of a 2nd order CRF (on the left) as a FST (on the right)

normalized. Instead, the weight of the whole path is normalized which alleviates the label-bias problem.

The associated undirected graph of a CRF also indicates the conditional independence assumptions of the models. In undirected graphs, independence can be established simply by graph separation: if every path from a node in X to a node in Z goes through a node in Y , we conclude that $X \perp Z|Y$. In other words, X and Z are independent given Y. Properly defining conditional in-dependencies is essential in any statistical machine learning application, as having too many parameters will most often result in degraded performance.

4.1 Why CRF ?

CRF is compatible with the nature of the problem: CRF is used for computing probability of label combinations of the whole sentence while other methods optimize word by word instead of the whole sentence, it causes CRF optimize probability better

than others, since the results from CRF are more consistent sentence-wise. Robust to over-fitting problem: Since we determine the structure of undirected graph and the structure is fixed (i.e.) There is no structural learning involved so the probability of over-fitting is less) Label Bias problem is solved, early words with low entropy (of $p(w_i|w_{i-1})$) do not cause bias for the solution. We can always find the global optimum of the Likelihood function. CRF is a well-known model primarily used in sequential classification problems. We can also think of a CRF as a finite state probabilistic transducer with un-normalized transition probabilities. However, unlike some other weighted finite-state approaches CRFs assign a well-defined probability distribution over possible labeling of a sentence, trained by maximum likelihood, which corresponds to the maximum entropy solution for CRF. Furthermore, the loss (negative log likelihood) function is convex, guaranteeing convergence to the global optimum. CRFs also generalize easily to analogues of stochastic context-free grammars. The transitions leaving a given state compete only against each other, rather than against all other transitions in the model. In probabilistic terms, transition scores are the conditional probabilities of possible next states given the current state and the observation sequence. This per-state normalization of transition scores implies a "conservation of score mass" whereby all the mass that arrives at a state must be distributed among the possible successor states.

In MEMMs, an observation can affect which destination states get the mass, but not how much total mass to pass on. The critical difference between CRFs and MEMMs is that a MEMM uses per-state exponential models for the conditional probabilities of next states given the current state, while a CRF has a single exponential model for the joint probability of the entire sequence of labels given the observation sequence. Therefore, the weights of different features at different states can be traded off against each other. We do not face label bias problem in HMM the structure is not fixed it has a tendency to over-fit A generative model (models p(X,Y)) CRF or MEMM are instead discriminative models (modeling p(Y|X)) Sensitive to marginal distribution of X, thus if X in test data are distributed different to X in training data, the performance suffers. CRF and MEMM are robust to this mismatch.

5. CONDITIONAL RANDOM FIELDS INFERENCE WITH VITERBI ALGORITHM

In this chapter we will explain the Conditional Random Fields (CRF) method applied in machine learning. We use CRF in training model. Also inference in Conditional Random Fields is done using the Viterbi algorithm an efficient dynamic programming algorithm.

5.1 Constraint Viterbi Algorithm For CRF

In supervised learning we learn parameters from data and inference from models using algorithms such as Viterbi. To be able to solve inference problems in CRFs, we need to be able to calculate the most likely label sequence (y^*) using the conditional probabilities ($p(y'_{1:T}|x_{1:T},\mathbf{w})$, i.e. the sequence y given the sentence features x and the learned weights $\mathbf{w}$) as modeled by the CRFs:

$$y^*_{1:T} = argmax_{y'_{1:T}} p(y'_{1:T}|x_{1:T},\mathbf{w}) \tag{5.1}$$

and for the learning problem, we need to calculate the partition function ($Z(.)$)

$$Z(\mathbf{x}_{1:T},\mathbf{w}) = \sum exp\{\sum_{j=1} w_j F_j(\mathbf{x}_{1:T}, y'_{1:T})\} \tag{5.2}$$

Note that direct calculation of these two quantities is highly expensive due to exponential amount of all possible $y_{1:T}$ that is needed to be considered. after expanding the features:

$$y*_{1:T} = argmax_{y'_{1:T}} \sum_j^T w_j \sum_{t=1} f_j(y'_{t-1}, y'_t, \mathbf{x}_{1:T}, t) \tag{5.3}$$

Let

$$g_t(y_{t-1}, y_t) = \sum_j w_j f_j(y_{t-1}, y_t, \mathbf{x}_{1:T}, t) \tag{5.4}$$

to simplify notation. Define partial maxima:

$$V(y,t) = max_{y'_{1:t-1}}(\sum_{\tau=1}^{t-1} g\tau(y'_{\tau-1}, y'_\tau) + g_t(y'_{t-1}, y)) \tag{5.5}$$

Clearly, this leads to a recursion: $V(y,t) = max_{y'}(V(y',t-1) + g_t(y',y))$ Similar to HMMs, we need to hold a back-pointer to the maximizer label (state) after each time step also we can view this procedure in a trellis. In the end, we can trace back from $V(y*_T, T)$ to obtain the most likely label sequence $y*_{1:T}$ We need to sum over exponentially many sequence labelings which is impractical. We can perform a dynamic programming algorithm like that to compute Z. We need to use the local features summed over time to do that:

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \sum_{y'_{1:T}} exp \sum_{\tau=1}^{T} \sum_{j=1}^{N_f} w_j f_j(y'_{\tau-1}, \mathbf{x}_{1:T}, \tau) \tag{5.6}$$

$$Z(\mathbf{x}_{1:T}, \mathbf{w}) = \sum_{y'_{1:T}} \prod_{\tau=1}^{T} G_\tau(y'_{\tau-1}, y'_\tau) \tag{5.7}$$

Where we define partial sums up to time t.

$$\alpha(y,t) = \sum_{y'_{1:t-1}} (\prod_{\tau=1}^{t-1} G_\tau(y'_{\tau-1}, y'_\tau) G_t(y'_{t-1}, y)) \tag{5.8}$$

We can update $\alpha(y,t)$ by the following forward recursion :

$\alpha(y,t) = \sum'_y \alpha(y',t-1) G_t(y',y)$

Similarly we define backward partial sums:

$$\beta(y,t) = \sum_{y'_{t+1:T}} (G_{t+1} \prod_{\tau=t+1}^{T} G_{\tau+1}(y'_\tau, y'\tau + 1)) \tag{5.9}$$

which can be updated with the backward recursion

$$\beta(y,t) = \sum_{y'} \beta(y',t+1) G_{t+1}(y,y') \tag{5.10}$$

We will cover the problem of inferring the most-likely state sequence given a CRF and an observation sequence.

If we use Viterbi without any constraint Viterbi consider all slot paths that were learned from the train data. for example for the following sentence if we consider only slot 1 :

- bu bu+Det (A_1) bu+Pron+DemonsP+A3sg+Pnon+Nom (A_2)
- yıl yıl+Noun+A3sg+Pnon+Nom($B1$) yıl+Verb+Pos+Imp+A2sg (B_2)

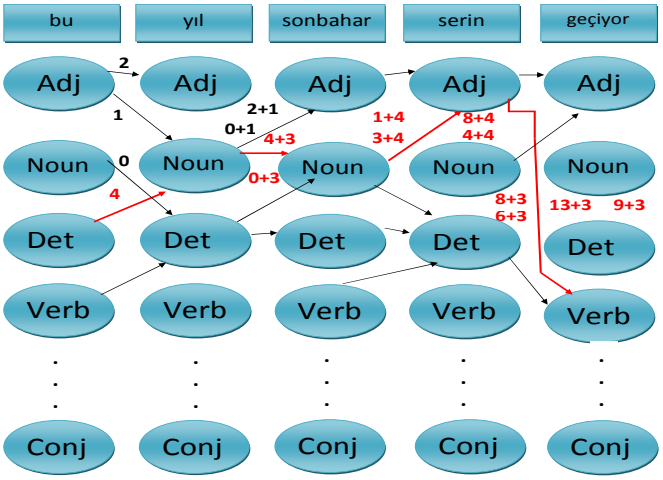

Figure 5.1: Viterbi paths for slot 1 values

- sonbahar sonbahar+Noun+A3sg+Pnon+Nom(C_1)
- serin serin+Adj($D1$) ser+Noun+A3sg+Pnon+Gen($D2$) ser+Noun+A3sg+P2sg+Nom($D3$) ser+Verb+Pos+Imp+A2pl($D4$) seri+Noun+A3sg+P2sg+Nom($D5$)
- geçiyor geç +Verb+Pos+Prog1+A3sg($E1$)

Viterbi paths are shown in Figure 5.1 :

As shown, Viterbi has to compute the weight's of all paths, if we consider that we have 9 slots and 116 tags, Viterbi repeats this huge calculating for 9 slots. We use constrained Viterbi in order to restrict the paths. In this case Viterbi considers the possible paths that are selected from morphological analysis results created by the morphological analyzer. In Figure 5.2 (A...E) are possible analysis results. All analysis results consist of 9 different slots. For example, analysis A_1 consists of (Det, Pron) for slot 1. As show in Figure 5.1, we eliminate the remaining slot 1 possible values since they are not valid tags indicated by the morphological analyzer.

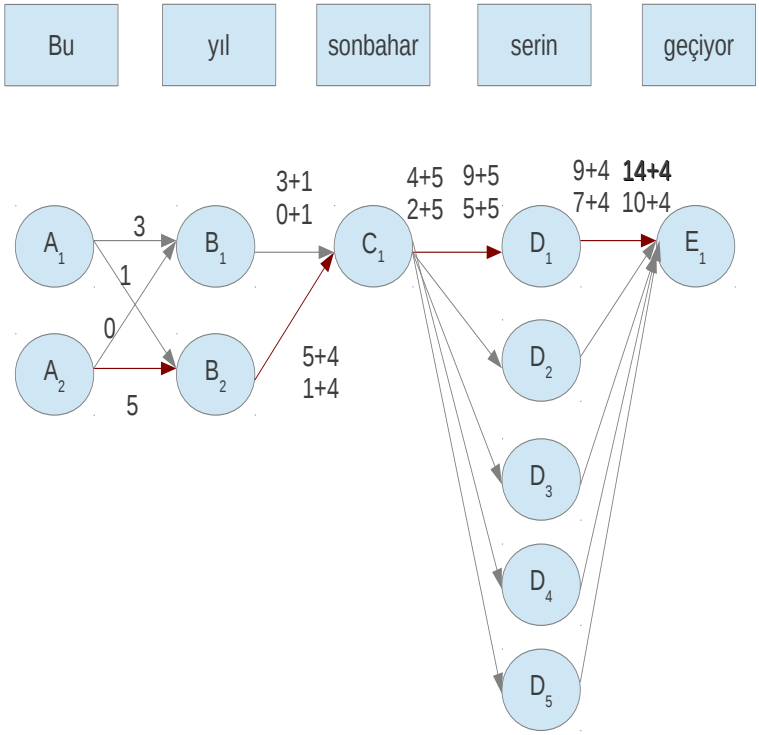

Figure 5.2: Inferring the most-likely state sequence

6. MORPHOLOGICAL DISAMBIGUATION USING CRFS

In this section we discussed methods for POS tagging problem and morphological disambiguation problem. In the proposed method, POS tagging of a sentence is performed in a series of steps. In the most basic form we begin by calculating the features related to the sentence, later the conditional probabilities of possible tag assignments are calculated and the most probable tag sequence are selected. The proposed method makes use of the MALLET library [16] and the mRMR source code found in [17].

6.1 POS Tagging

In this section we discuss the basic approach for POS tagging and the several modifications to improve efficiency and/or performance.

6.1.1 Basic model

The CRF trained for POS tags are conditioned on the features of the sentence. However, during POS tagging, we also know a set of possible tags given by the morphological analyzer, which we call possible solution sequences ($\mathbf{S}_i$). Thus, we have a further conditioning.

$$p(\mathbf{S}_i|C) = \frac{p(\mathbf{S}_i|\mathscr{F}(C))}{\sum_j p(\mathbf{S}_j|\mathscr{F}(C))} \tag{6.1}$$

Where C is the sentence and $\mathscr{F}(C)$ is the corresponding feature representation of the sentence, as given to the CRF. In other words, we do not assign the most probable tag sequence according to the conditional probability given by the CRF but select the most probable sequence ($\hat{\mathbf{t}}$) among possible sequences instead. This selection is performed by a constrained Viterbi approach, where the Viterbi is run on states that are deemed

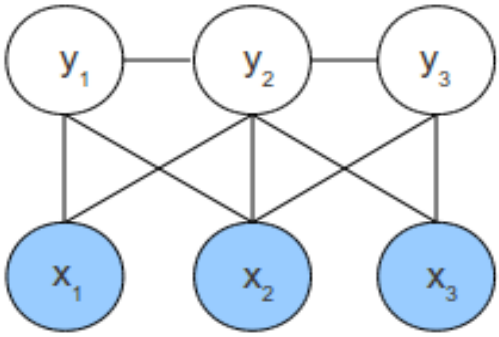

Figure 6.1: The graphical model of the proposed approach

possible by the morphological analyzer, instead of running Viterbi on the whole state space.

$$\hat{\mathbf{t}} = \arg\max_{S_i} p(\mathbf{S}_i|C) \tag{6.2}$$

The graphical model for the proposed method is shown in Figure 6.1.

Figure 6.2 shows a sample sentence and how our method chooses the POS tags. The top part of the figure shows the features for the respective words and the bottom part shows the possible POS tags as given by the analyzer. The values indicated above the arrows show transition weights. Note that in this example, any path from a tag of the initial word to a tag of the last word is a possible solution. In this figure, the weights of the transitions are the functions of the initial state, the final state and the features of the final word. The weight function is actually a factored expression, where

$f(s_i, s_{i+1}, \mathscr{F}(w_{i+1})) = q(s_i, s_{i+1})q(s_{i+1}, \mathscr{F}(w_{i+1}))$,

the first term corresponds to the edge features and the second term corresponds to node features.

6.1.2 Alternative models

The basic approach of using CRF for POS tagging has an important disadvantage: high computational complexity. To remedy this issue, we propose these methods: dividing sentences into shorter sub-sentences and using marginal probabilities of tag assignments per word to eliminate the unlikely tags. In addition, we introduce a new approach to improve the performance of the basic method without significant overhead.

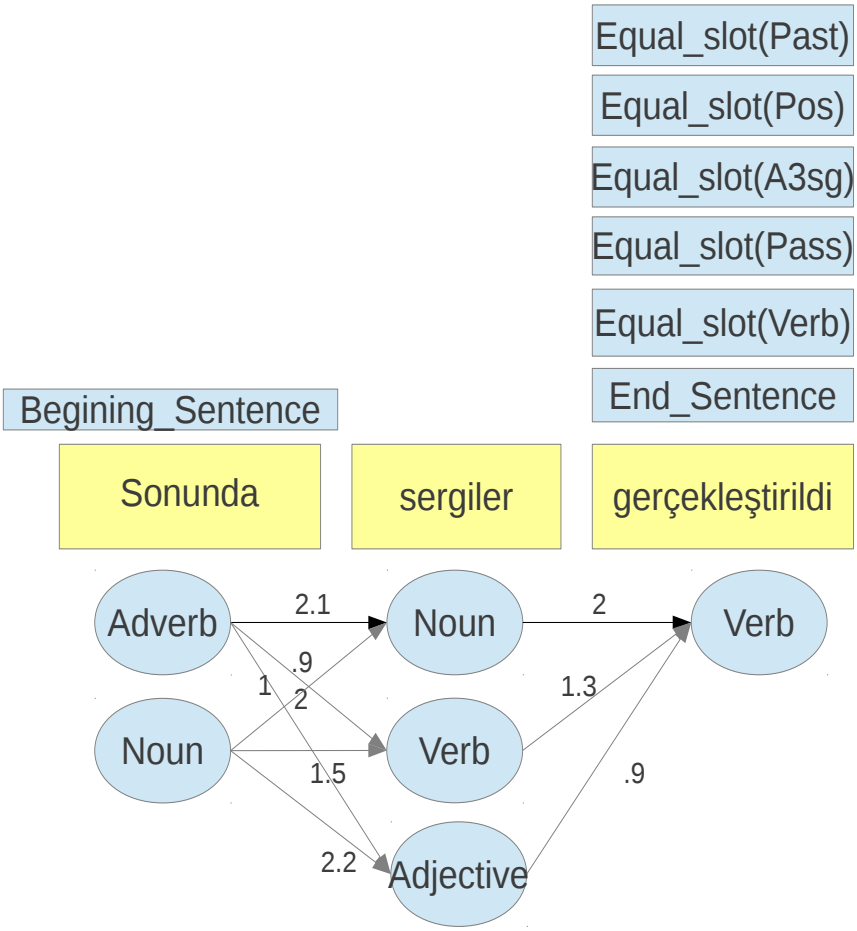

Figure 6.2: A sample sentence ("The exhibition has been finally realized.") with features and possible solutions. The tag chosen by our method is shown in bold arrows.

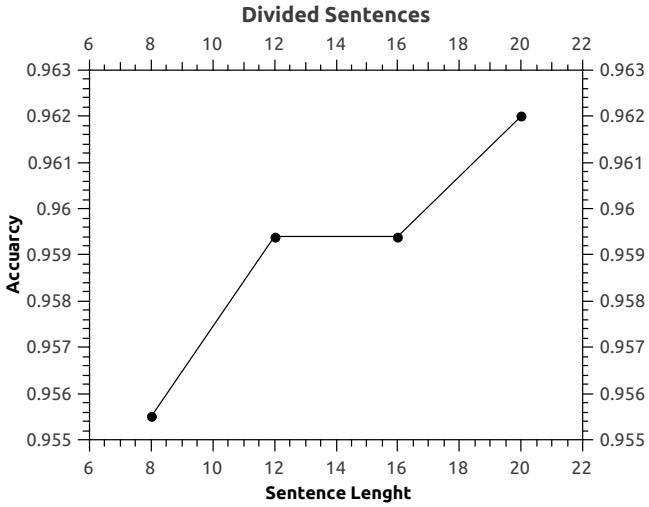

Figure 6.3: Accuracy vs. the length of the partial sentences

In this section, we describe these methods and briefly comment on their performances. The quantitative results will be given in the Results Section.

Note that the complexity of the constrained Viterbi is $O(T \times |S|^2)$, where T is the length of the sequence and $|S|$ is the maximum number of possible states in any element of the sequence.

6.1.2.1 Model I: splitting sentences

This fast approximation method is conceptually the easiest one. The idea is to split a long sentence into multiple parts such that each part is shorter than a maximum length. Let's explain this method with an example sentence from our corpus. This sentence has 35389440 different possible morphological analysis sequences. The poor performance that would result from calculating the probabilities of all of these possible solutions is obvious. Now suppose we divide the sentence into 4 parts of lengths 9, 9, 9, 7. The corresponding number of possible solutions are 384, 960, 30 and 32 which sum up to 1406. The huge savings in the number of solutions to consider is apparent. However, despite these good reductions in the number of possible solutions to consider, this method results in the worst accuracy among the alternatives. This is due to the fact that splitting sentences this way enforces an independence assumption on the splitted sub-sentences, which reduces the performance especially in words that are closer to the cut-off boundaries. The Figure 6.3 shows the trade off between the performance and the length of the partial sentences.

Using this approach, the complexity of disambiguating a sentence is reduced to $O(T' \times |S|^2)$, where T' is the maximum length of the sub-sentences, so the reduction is linear.

6.1.2.2 Model II: trim unlikely tags

Notice that the complexity of the constrained Viterbi is linear on the length but quadratic on the maximum number of states for any element of the sequence. This observation becomes even more important when we note that the number of possible analysis of a word can reach up to 23 in our corpus and possibly more in general texts. Thus a reduction on the number of possible tag assignments of a word can have significant effects. Out of the many possible sequences for the sentence mentioned in Section 6.1.2.1, many include highly unlikely values for some words. The approach discussed in this section exploits this pattern by trimming out the highly unlikely tags for words but still allowing multiple possible POS tags. In our implementation, we select the words for which the number of possible tag assignments is greater than 6. For such words, we remove the least likely tag assignments using marginal probabilities until either this number is 6 or the number of eliminated tags is 5. We use such an upper limit in order not to remove too many such tags in order not to degrade accuracy. The additional complexity of this approach is obviously linear on the length of the sequence and the trimmed sequence can be disambiguated by constrained Viterbi in $O(T \times 6) = O(T)$. We can see that there can be huge savings in long sentences with complex morphological properties. The conservative approach outlined here means the accuracy is not effected at all, as shown in the next section.

6.1.2.3 Model III: model complexity of the solutions

An interesting observation of morphological properties of words in Turkish is that the correct POS tags of the words tend to be the less morphologically complex ones. In other words, simpler interpretations of words tend to be used more often than the more complex ones of the same word. One way to operationalize this observation is to take the Bayesian stance and model a prior. However, correctly assigning numerical values for our prior knowledge is difficult and we take the other position, where the nature of this relation is learned from the data itself. In Turkish, the morphological complexity of a word can be modeled by the number of IGs of it. Thus we model this number with

a 0-order CRF, since we do not expect the neighboring IG counts to effect each other. This CRF is combined with the original one by multiplying the probabilities, i.e. we assume the number of IGs and the POS tags to be independent, which is reasonable. Since we use a 0-order CRF, the complexity of inference is only $O(T \times |S|)$. However, we do note increased performance as can be seen in the next section.

6.2 Morphological Disambiguation

In this section we discuss the basic approach for morphological disambiguation and the several modifications to improve efficiency and/or performance.

6.2.1 Combine CRFs

Before we discuss about full morphological disambiguation problem, we want to discuss how we combined CRFs. We already mentioned that we change our problem to multiple class classification problem. This means that we have one CRF for one slot. Each CRF solve disambiguation problem of the corresponding slot. For full morphological disambiguation, we combine these CRFs. In figure 6.4 we discuss how we get transition weight from each CRF. We show possible morphological analysis results for each word with the letters A,B and C. In our sample sentence, each word has 3 different morphological analysis results, we want to find the transition weights between A, B and C. For example for finding the transition weight between A1 and B2 we must get all transition weights between A1's slot values and B2's slot values. In this case we show only 4 CRFs that determine slot1, slot3, slot4 and slot5. For calculating the transition weight between A1 and B2, we sum up all transition weights coming from different slot CRFs. When all paths are weighted, Viterbi algorithm selects the most-likely morphological analysis results sequence. We discuss how Viterbi algorithm works in chapter 5. When we deal with the POS tagging problem, we also used Viterbi algorithm for calculating the weight's of all paths. Viterbi algorithm in the POS tagging problem find the most probable tag sequence. Finally, this idea of combining several CRFs to solve the morphological disambiguation problem is shown in Figure 6.5. This figure shows how the input given by the morphological analyzer is turned into a feature representation by predefined feature templates, after which the

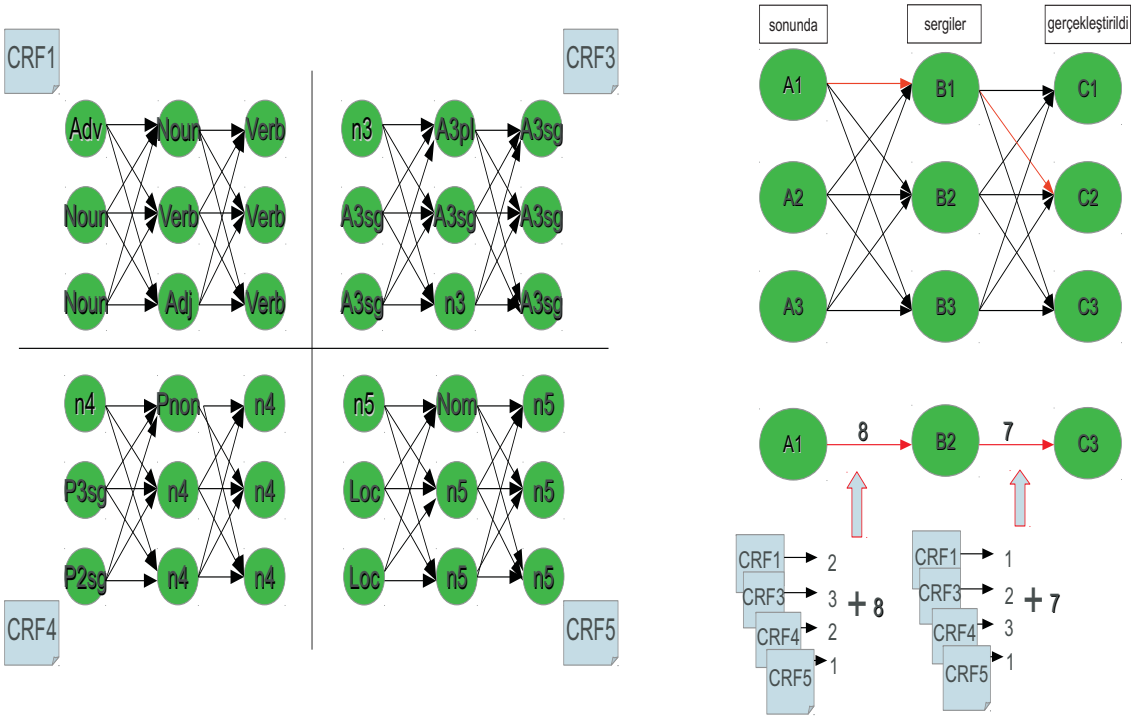

Figure 6.4: Overviwe of combining multiple CRFs

individual probabilities assigned by the CRFs are combined together with the score from the root Ngrams and the maximum scored morphological analysis sequence is chosen.

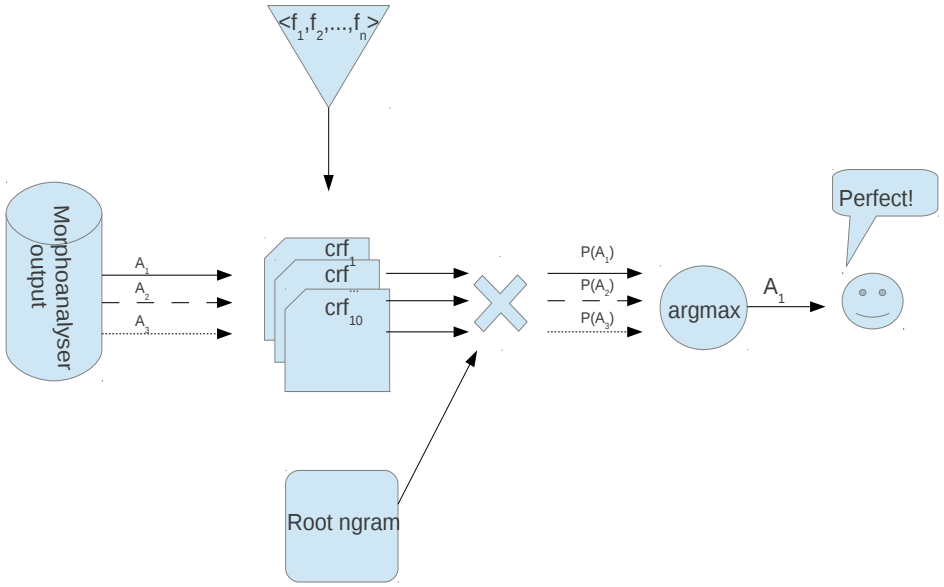

Figure 6.5: Combining multiple CRFs and ngram model

6.2.2 Basic model

The major problem with trying to estimate the probability of a particular morphological parse sequence ($\mathbf{m}$) for the sentence (S), that is $p(\mathbf{m}|\mathscr{F}(S))$, is the enormous number of possible values for $\mathbf{m}$. This big number, together with the available NLP corpora, means the joint estimates for the tags will not be reliable. Our approach is to partition the morphological parses $\mathbf{m}$ into tag sequences $(t_1, t_2, \ldots, t_9)$ such that each tag for j'th word $t_{i,j}$ will take on values from disjoint sets of tags T_i and these random variables will be assumed to be conditionally independent, i.e. $p(\mathbf{m}|\mathscr{F}(S)) = \Pi_{i=1..9} p(t_i|\mathscr{F}(S))$. We call the set of possible values T_i as slots and the corresponding values as slot values.

The necessity of assuming a structure for $\mathbf{m}$ is obvious, as otherwise effective estimation will not be possible. However, the particular assumption in our model may still be questioned. However, when we note that the independence assumption is a conditional one, we see that the dependence of slots are still modeled when we take the distribution on input into account. The slots themselves are determined by requiring that no analysis can take multiple different tags from one slot and the slots

themselves should have a common semantic interpretation. This can also be seen as another justification of local independence assumption. These two requirements lead us to design the slots as shown in Table 2.1.

Given this slot structure, our approach is to model each slot using a Conditional Random Field. The estimation of model parameters per slot is much less problematic than that of the joint distribution, and we shall demonstrate that this model can disambiguate the morphological tags with a very high success.

6.2.3 Improving efficiency

The main concern for the practical application of CRFs in the literature is its efficiency issues. Many authors mention the high complexity of the inference step [18]. In this section, we shall discuss several schemes to improve the efficiency, without reducing the performance as much as possible. The schemes that we will discuss are; dividing sentences, selecting a subset of minimally sufficient identifying tags (distinguishing markers list) and trimming the solution space.

To facilitate the comparison, in all of the coming discussion, we shall employ the following sentence as an example (an excerpt from an Omar Khayyam's poem): "Tanrıya toz kondurmamak meleğin işi olsun ve temizlik, cennet kapıcısının işi" (Let the angels try to keep god from blame and the doorkeeper of the heavens do the cleansing). The number of possible morphological analysis results for per word of the sentence is 2, 1, 4, 3, 4, 5, 1, 3, 2, 5, 4. The number of possible solution sequences is $2 \times 1 \times 4 \times 3 \times 4 \times 5 \times 1 \times 3 \times 2 \times 5 \times 4 = 57600$

6.2.3.1 Dividing sentences

One of the simplest possible ideas that also is described in 6.1.2.1 is to split the sentences into non-overlapping sub-sentences to be disambiguated independently. Of course, since we know the sub-sentences are actually parts of a larger sentence, we can still keep the same features for the sub-sentences as in the original sentence. This splitting procedure can drastically reduce the number of possible sequences.

Even though we have a very good reduction in terms of probability evaluations, this method also performs the worst. As we will show in the Experimental Results section, the reduction in performance is too large for this approach to be usable.

6.2.3.2 Distinguishing markers list

In order to fully disambiguate a sentence we do not need to know the values for all slots. To calculate a minimal number of slots to fully disambiguate a sentence, we employ a greedy mechanism. In this approach, we iteratively search for the slot which decreases the ambiguity most when the correct value of the slot is known. We keep

adding slots to our distinguishing marker list (DML) in this fashion, until there is no ambiguity left.

The ambiguity is measured using entropy. In order to calculate the entropy of a sentence, we first calculate the entropy of words. The entropy of a word is calculated by assuming that the correct analysis is distributed uniformly over the set of possible solutions. The decrease in ambiguity when the correct analysis for slot i is known is calculated using mutual information:

$$MI_w(Y_w; Y_w^i | Y_w^S) = H(Y_w|S) - H(Y_w^| Y_w^i, Y_w^S) = -log(f_w^S) + log(f_w^{(S \cup i)}) \quad \textbf{(6.3)}$$

On the above equation, vector random variable Y_w denotes the correct morphological analysis of word w, while Y_w^j denotes the jth component of the correct analysis. The set S is the DML at the current step. To determine the slot to add to DML, we sum up the $MI_w()$ for all words in the sentence. The slots to be added to DML are determined by:

$$s =_j \sum_w MI_w(Y_w; Y_w^j | Y_w^{S^t}) S^{t+1} = S^t \cup s \quad \textbf{(6.4)}$$

The procedure is repeated until MI for all remaining slots are 0.

The advantage of determining DML is to increase the efficiency of the procedure, as running all 9 CRF classifiers for each sentence is time consuming. The drawback, however, is the degraded performance. One can improve the results by taking an alternative approach, where the slots are added to DML two by two instead of one by one. In this way, the decisions of different classifiers better support each other and higher accuracy is obtained as we will discuss in the Experimental Results section.

Considering the example sentence, the DML approach gives slots 1,2, and 4 as the minimal distinguishing markers. This reduces the number of probability evaluations of 518400 in the original problem to 172800. As we can see, the reduction is not as impressive as in the previous case. An example of how DML works maybe found in the Appendix Section.

6.2.3.3 Word-Wise trimming unlikely solutions

This approach that we described in section 6.1.2.2 is the most adaptive one, since by keeping the trimming limit high, we have a very good performing method, while on the other hand, keeping the trimming level low (such as 500 or 100) we have a very fast method that still compares well with other alternatives. Suppose we trim the number of solutions per word to 2. Our sentence has length 11 and 9 of the words have ambiguity This leaves us, after trimming, 2^9, or 512, possible solution sequences .If we consider the aforementioned sentence, the first step reduces the number of probability evaluations to $512 \times 9 = 4608$, which may be reduced further using the second step. This approach is the best performing one out of the three approaches when the trimming limit is reasonable (say 5000 or 10000).

6.2.4 N-gram model

An n-gram model is a type of probabilistic language model for predicting the next item in such a sequence in the form of a $(n-1)$–order Markov model. An n-gram model models sequences, including natural language sequences, using a Markovian approach. More concisely, an n-gram model predicts r_w based on $r_{w-(n-1)}...r_{w-1}$. In probability terms, this is modeling $P(r_w|r_{w-(n-1)},..,r_{w-1})$. When used for language modeling, independence assumptions are made so that each word depends only on the last $n-1$ words. This Markov model is used as an approximation of the true underlying language. This assumption is important because it massively simplifies the problem of learning the language model from data.

We use n-gram approach to model root sequences. We choose n-grams rather than a more complex scheme such as CRFs or HMMs because there are many roots leading to huge state spaces, which makes inference in such models slow. Our approach to incorporate n-grams into the CRF model proposed is to factor in the score given by n-grams to the joint probability given by the multiple CRFs.

We add root n-gram model for word w to our formula introduced in basic model as follows:

$$P^{'}(\mathbf{t}|\mathscr{F}(S)) = \Pi_{t_i \in \mathbf{t}} P(t_i|\mathscr{F}(S)) p(r_i|r_{i-1})] \tag{6.5}$$

where r_i denotes the root of i and $p(r_i|r_{i-1})$ is the conditional probability estimate from the n-gram model. Here, we assume $p(r_0|r_{-1}) = p(r_0)$, i.e. for the first word, the probabilities are estimated unconditionally. As one can see, this scheme simply reweighs the original CRF probabilities, thus including root consistency to the model which, in turn, results in a slight performance improvement. We used Srilm [19] for creating Language Model.

7. EXPERIMENTAL RESULTS

In this section, we first show the effect of feature selection on the performance. We then show the performance of the proposed method on a common dataset and compare it with the method of [8], which is considered as the state of art. The results are obtained using default parameters of the mallet library. The Java source codes used in the experiments will be made available online.

7.1 POS Tagging Results

The results for the proposed method, together with the results from [8] (Perceptron) are given in Table 7.1. We use the same training data (1 million words) that is used in these studies. The training data is a semi-automatically tagged data set which consists some erroneous analysis results. In this study, we strived to correct as many errors as possible and trained our methods as well as the previous methods on this dataset. We have also accounted to the difference in tags employed in Hasim Sak's method and ours so we kept two separate training files, each having the same corrections but slightly different tags, so that Hasim Sak's method does not suffer from the changes in some of the tag names. Our test data (a manually disambiguated data consisting nearly 1K words) is again from [20]. Note that this set also contains erroneous analysis results, which we had to correct. All the results are reported using this corrected dataset, which will be made available to researchers. These corrections are the reason why our results are slightly different than the ones reported in [8] The results are reported in Table 7.1.

The results in Table 7.1 exclude the punctuations in computing the accuracy. The results indicate the competitiveness of our approach. It is important to recognize that the POS tagging in Perceptron [8] method is performed by selecting the appropriate tags after a full morphological disambiguation. On the contrary, our method directly assigns a POS tag sequence to the sentence. The output of our method need not be a single assignment, instead we can output different "belief levels" for different tag

Table 7.1: Pos Tagging Performances

Method	test set
Perc [8]	98.60
Basic Model	98.35
Model I	96.2
Model II	98.35
Model III	98.60
Model II + Model III	98.60

assignments. If these POS tags are to be used in another procedure as an intermediate step, this will also be an advantage. Finally, the method in [8] contains a lot more number of features than our proposed approach, since our approach is flexible in the selection features, it can be extended using additional features from the Perceptron method.

7.2 Automatic Feature Selection Results

Feature selection is an important step in many machine learning tasks. The effect of feature selection is two-folds, the reduction of features may actually increase classification performance, since accidental correlations in the training data can mislead the classifier and generalization capability of classifiers is expected to be better for lower model complexity. Another effect is the improvement in training and classification efficiency, since inference in the model with a fewer number of features will be faster. For these reasons, we have dismissed the features that are not selected in the top 230 by mRMR.

Figure 7.1 shows the accuracy vs. the number of features. We can see that reducing the features below 230 degrades the performance significantly. Even though a significant increase in performance is not observed for the particular validation set, the reduction in features is still relevant to reduce computational complexity in test and training.

7.2.1 Constrained Viterbi vs. linear search

When we use Linear Search instead of Constrained Viterbi we faced with unpractical problem. Duration of POS tagging in this case for higher sentence length is up to one day. Because of this, we use trim unlikely tags for restrict sentence lengths. We explain

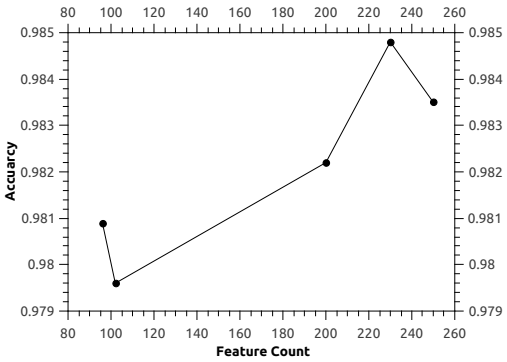

Figure 7.1: Accuracy vs. number of features selected by mRMR

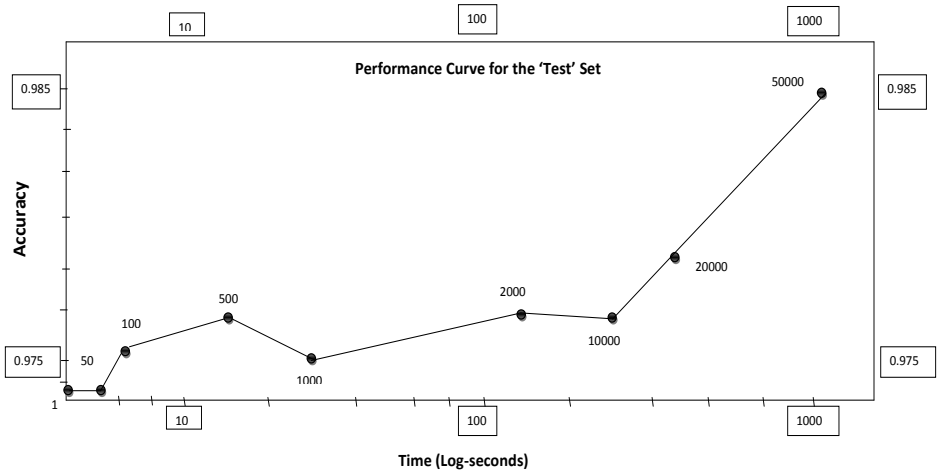

Figure 7.2: Accuracy vs. run time using linear search and word-wise tag trimming

this method in section 6.1.1.4. The Figure 7.2 shows the run time vs. performance trade-off using this approach. Note that in this figure, the x-axis shows logarithmic time. If we use Viterbi instead of Constrained Viterbi in POS tagging problem we face with the selected analysis results that are not in legal morphological analysis results which are output of the morphological analyzer. In POS tagging problem, using Viterbi without any restriction, cause we reselected analysis results between legal analysis results.

7.3 Disambiguation Results

In this section, we show the disambiguation results for different models and compare them. We used two test sets, because, first test data set is very small but the second one, being bigger, is also more noisy and has incorrect tags. Our tests have different versions, resulting from different combinations of CRFs and Ngrams. For example,

Table 7.2: MD Performances

Method	**test set A**	**treebank test set**
without any lexical information model	93.5	85.60
using stem information only	94.67	86.00
using surface form of word	95.57	86.47
using surface and stem information	95.30	86.84
Perceptron method	96.31	88.17

we have observed that not using the CRF for second group improved performance in the small test set, but degraded in the bigger set. We also try different feature sets that exploits different properties of the Turkish language. We also tried incorporating slots from IGs other than the last to see if it improved disambiguation. In this experiment we used only slot 1, slot 3, slot 5 that we think carry important information. But we concluded that these do not posit significant improvements. Finally, we experimented with weighing the CRF scores (the conditional probabilities) when bringing them together to get the final probability value, but did not notice any improvement so we do not report these experiments.

In Table 7.2, we show how our model performce under various settings in the two test data sets vs. the Perceptron method [8].

8. CONCLUSIONS

In this thesis, we proposed a method using Conditional Random Fields to solve the problem of POS tagging and morphological disambiguation in Turkish. We have shown that using several features derived from morphological and syntactic properties of words and feature selection, we were able to achieve a performance competitive to the state of art. Furthermore, the probabilistic nature of our method makes it possible for it to be utilized as an intermediate step in another NLP task, such that the belief distribution can be used as a whole instead of a single estimate. Note that our proposed method can also be employed to other languages, perhaps with the addition of language dependent features.

Another major contribution of this work is the discussion on several approaches to improve efficiency of POS tagging and disambiguation using CRFs. We have shown some simple heuristics that can improve inference performance and efficiency. We believe this work constitutes a major step towards making CRF a more practical tool in NLP.

As part of our future work, we plan to investigate the addition of other features to improve the performance of the proposed method. One possibility is to incorporate features based on lemmatization.

REFERENCES

[1] **Smith, N.A.**, **Smith, D.A. and Tromble, R.W.**, 2005. Context-based morphological disambiguation with random fields, Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05, Association for Computational Linguistics, Stroudsburg, PA, USA, pp.475–482, http://dx.doi.org/10.3115/1220575.1220635.

[2] **Sutton, C. and McCallum, A.**, 2010. An introduction to conditional random fields, *Arxiv preprint arXiv:1011.4088*.

[3] **Lafferty, J.**, **McCallum, A. and Pereira, F.C.N.**, 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

[4] **McCallum, A. and Li, W.**, 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons, Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003 - Volume 4, CONLL '03, Association for Computational Linguistics, Stroudsburg, PA, USA, pp.188–191, http://dx.doi.org/10.3115/1119176.1119206.

[5] **Kudo, T.**, **Yamamoto, K. and Matsumoto, Y.**, 2004. Applying conditional random fields to Japanese morphological analysis, Proc. of EMNLP, volume2004.

[6] **Shacham, D. and Wintner, S.**, 2007. Morphological disambiguation of Hebrew: A case study in classifier combination, Proceedings of EMNLP-CoNLL, volume 7, pp.439–447.

[7] **Habash, N. and Rambow, O.**, 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop, Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, ACL '05, Association for Computational Linguistics, Stroudsburg, PA, USA, pp.573–580, http://dx.doi.org/10.3115/1219840.1219911.

[8] **Sak, H.**, **Gungor, T. and Saraclar, M.**, 2007. Morphological Disambiguation of Turkish Text with Perceptron Algorithm, Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Text Processing, CICLing '07, Springer-Verlag, Berlin, Heidelberg, pp.107–118, http://dx.doi.org/10.1007/978-3-540-70939-8_10.

[9] **Peng, H.**, **Long, F. and Ding, C.**, 2005. Feature selection based on mutual information criteria of max-dependency, max-relevance, and

min-redundancy, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **27(8)**, 1226 –1238.

[10] **Mengusoglu, E. and Deroo, O.**, 2001. Turkish LVCSR: Database Preparation and Language Modeling for an Agglutinative Language, in ICASSP'2001, Student Forum, Salt-Lake City.

[11] **Hakkani-Tür, D.**, **Oflazer, K. and Tür, G.**, 2002. Statistical Morphological Disambiguation for Agglutinative Languages, *Computers and the Humanities*, **36(4)**, 381–410, http://www.springerlink.com/content/p8217051382887pm/abstract/.

[12] **Oflazer, K.**, 1995. Two-level Description of Turkish Morphology, *Literary and Linguistic Computing*, **9(2)**, 137–148, http://llc.oxfordjournals.org/content/9/2/137.full.pdf+html.

[13] **Guyon, I. and Elisseeff, A.**, 2003. An introduction to variable and feature selection, *J. Mach. Learn. Res.*, **3**, 1157–1182, http://dl.acm.org/citation.cfm?id=944919.944968.

[14] **Gulgezen, G.**, **Cataltepe, Z. and Yu, L.**, 2009. Stable and Accurate Feature Selection, **W. Buntine**, **M. Grobelnik**, **D. Mladenić and J. Shawe-Taylor**, editors, Machine Learning and Knowledge Discovery in Databases, volume5781, chapter 47, Springer Berlin Heidelberg, Berlin, Heidelberg, pp.455–468, http://dx.doi.org/10.1007/978-3-642-04180-8_47.

[15] **McCallum, A.**, **Freitag, D. and Pereira, F.C.N.**, 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation, Proceedings of the Seventeenth International Conference on Machine Learning, ICML '00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp.591–598, http://dl.acm.org/citation.cfm?id=645529.658277.

[16] **McCallum, A.K.**, 2002. MALLET: A Machine Learning for Language Toolkit, http://mallet.cs.umass.edu.

[17] **Peng, H.**, 2012, mRMR (minimum Redundancy Maximum Relevance Feature Selection), http://penglab.janelia.org/proj/mRMR/.

[18] **Sutton, C.**, **Mccallum, A.**, **Getoor, L. and Taskar, B.**, 2006. Introduction to Conditional Random Fields for Relational Learning, Introduction to Statistical Relational Learning, MIT Press.

[19] **Stolcke, A.**, 2002. SRILM - An Extensible Language Modeling Toolkit, pp.901–904.

[20] **Yuret, D. and Töre, F.**, 2006. Learning morphological disambiguation rules for Turkish, Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, HLT-NAACL '06, Association

for Computational Linguistics, Stroudsburg, PA, USA, pp.328–334, http://dx.doi.org/10.3115/1220835.1220877.

APPENDIX A.1 : Distingutioning Marker List

Appendix A.1:HOW DML WORKS

In this section, we show an example for finding Distinguishing marker lists. The procedure is examplified step by step in the following figures.

In Figure A.1, we show the first step of the process, where we determine the slots of which knowing the correct tag results in the fewest remaining possible solution sequences. We have repeated the procedure in Figure A.2, which reduces the number of possible solutions even more. Finally, in the last figure we observe that the procedure ends when knowing the correct values for all slots in the Distinguishing Marker List results in the elimination of all ambiguity in the problem.

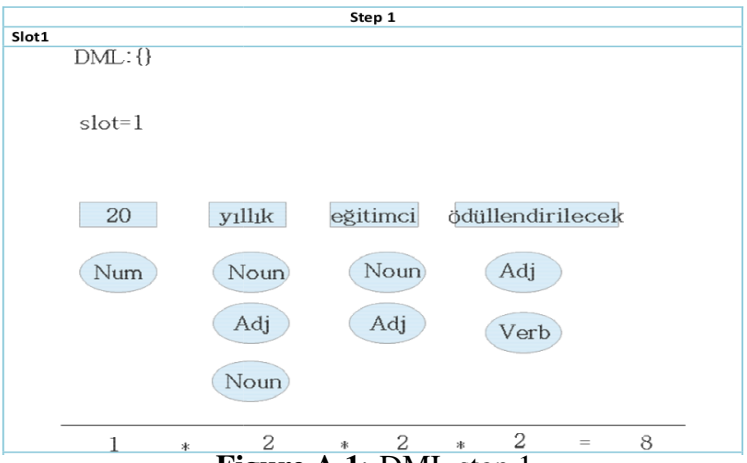

Figure A.1: DML step 1

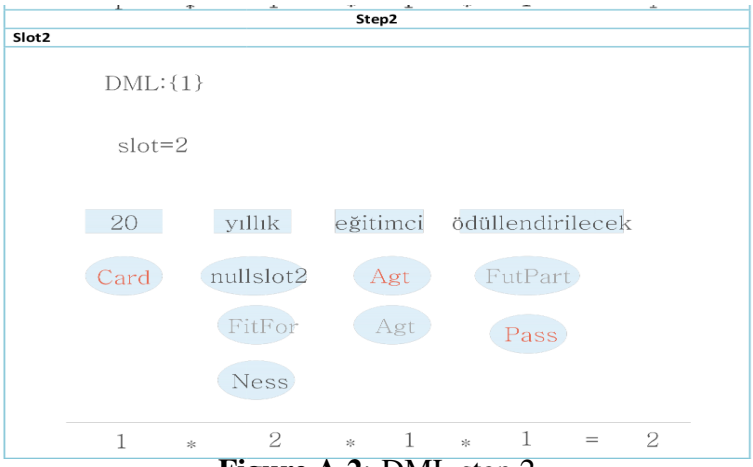

Figure A.2: DML step 2

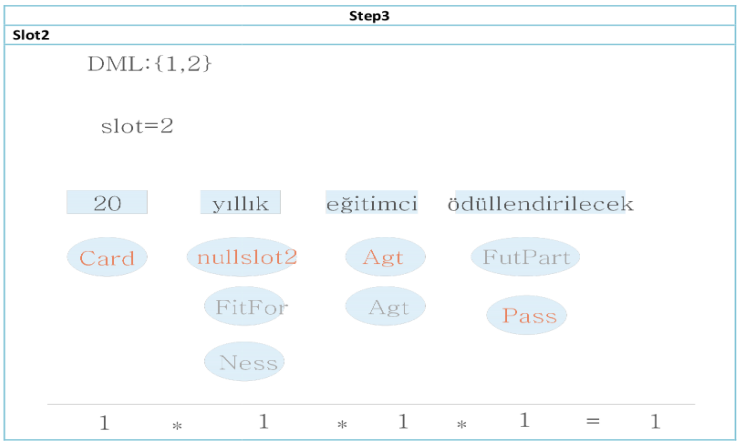

Figure A.3: DML step 3

CURRICULUM VITAE

Candidate's full name:	Razieh Ehsani
Place and date of birth:	Shabestar,Iran, 09 December 1984
Universities and Colleges attended	Istanbul Technical University
Address:	ITU Ayazaga Campus, Faculty of Computer and Informatics, Sarıyer, İstanbul
Phone:	0212 5029882
email:	rehsani@itu.edu.tr