DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

EXTRACTION OF NAMED ENTITIES FROM TURKISH DOCUMENT COLLECTIONS

by
Okan ÖZTÜRKMENOĞLU

November, 2018
İZMİR

EXTRACTION OF NAMED ENTITIES FROM TURKISH DOCUMENT COLLECTIONS

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy in Computer Engineering

by
Okan ÖZTÜRKMENOĞLU

November, 2018
İZMİR

PH.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"EXTRACTION OF NAMED ENTITIES FROM TURKISH DOCUMENT COLLECTIONS"** completed by **OKAN ÖZTÜRKMENOĞLU** under supervision of **ASSOC.PROF.DR. ADİL ALPKOÇAK** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

Assoc.Prof.Dr. Adil ALPKOÇAK

Supervisor

Asst.Prof.Dr. Semih UTKU

Thesis Committee Member

Asst.Prof.Dr. Güleser KALAYCI DEMİR

Thesis Committee Member

Doç. Dr. Deniz KILINÇ

Examining Committee Member

Doç. Dr. Alper ÖZCAN

Examining Committee Member

Prof.Dr. Kadriye ERTEKİN

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

Initially, I would like to emphasis my gratitude to my advisor Associate Professor Dr. Adil ALPKOÇAK for his valuable suggestions, encouragement, patience and support during this study. It was a great honor for me to work with him. In addition to his academic point of view, I have gained significant achievements in every aspect through his approach to events in life. He is always more than a supervisor for me.

I would like to thank my thesis tracking committee members Dr. Semih UTKU and Dr. Güleser KALAYCI DEMİR for their contribution to this study and sharing their ideas during development and writing of this thesis.

This research was supported under Contract Number: 2014.KB.FEN.018 (2013297) Dokuz Eylül University Scientific Research Projects (Bilimsel Araştırma Projeleri, BAP) Coordination Unit.

I have special thanks to my fabulous mother Binnaz, my father Mustafa, my sister Betül, my gorgeous wife Şule and my lovely daughter Sude as well as great friends Dr. Emel ALKIM, Dr. Hakan BULU, Dr. İbrahim ARPALIYİĞİT and Dr. Mete Uğur AKDOĞAN for their support and patience during the development and writing of the thesis.

Okan ÖZTÜRKMENOĞLU

EXTRACTION OF NAMED ENTITIES FROM TURKISH DOCUMENT COLLECTIONS

ABSTRACT

This thesis aims to develop a model improving Hidden Markov Model (HMM) and Conditional Random Field (CRF), which are two common sequence classifier techniques, for Named Entity Recognition (NER) task on Turkish documents. So, we first examined for the best values of parameters used as input in these models. In HMM, we represented each token with multi features. Next, we used CRF model to determine most effective parameters values that are used as input in this model such as window size, output encoding format and features extracted from tokens. After detailed examination of both HMM and CRF models, we applied a linear-chain CRF model, for NER in Turkish documents.

Besides, we proposed 41 different features in four categories: rule based, lexical, dictionary lookup and morphological based features. First, we performed a set of experiments using this feature set on publically available NER datasets. We achieved the best performance with a linear-chain CRF model using [-3, +3] as a window size, BIO encoding as an output encoding format and extended feature set. In terms of F1-measure, we obtained the 91.83 percent, 91.2 and 88.62 for person names, location names and organization names respectively.

Furthermore, this thesis also presents METU-NER corpus, which is based on annotation METU corpus for NER. We evaluated our a linear-chain CRF model with the same parameters used in the previous dataset. In terms of F1-measure, we achieved 73.26 percent, 70.12, 63.83, 61.54 and 69.14 for person, location, organization, temporal names and overall, respectively.

Keywords: Named Entity Recognition, Turkish NER corpus, Sequence Classifier Models, Information Extraction

TÜRKÇE DOKÜMAN KOLEKSİYONLARINDAN VARLIK İSİMLERİNİN ÇIKARIMI

ÖZ

Bu tez, Türkçe dokümanlarda Varlık İsmi Tanıma (VİT) görevi için iki yaygın dizi sınıflandırıcı teknik olan Saklı Markov Model (SMM) ve Koşullu Rasgele Alan (KRA)'ı iyileştiren bir model geliştirmeyi hedefler. Bu nedenle, ilk olarak bu modellerde girdi olarak kullanılan parametrelerin en iyi değerlerini inceledik. SMM'de her bir belirtkeyi çoklu özelliklerle temsil ettik. Daha sonra, KRA modelini, pencere boyutu, çıktı kodlama formatı ve belirtkelerden çıkarılan özellikler gibi bu modelde girdi olarak kullanılan parametrelerin en etkili değerlerini belirlemek için kullandık. Hem SMM ve hem de KRA modellerinin detaylı incelemesinden sonra, Türkçe dokümanlarda VİT için lineer zincirli bir CRF modeli uyguladık.

Ayrıca, dört kategoride 41 farklı özellik önerdik: kural tabanlı, sözcüksel, sözlük araması ve morfoloji temelli özellikler. İlk olarak, bu özellik kümesini kullanarak kamuya açık VİT veri setleri üzerinde bir dizi deney gerçekleştirdik. Pencere boyutu olarak [-3,+3], çıktı kodlama formatı olarak BIO kodlaması ve genişletilmiş özellik kümesini kullanarak lineer zincirli CRF modeli ile en iyi performansı elde ettik. F1 ölçütü olarak, sırasıyla kişi isimleri, yer isimler ve kurum isimleri için 91.83, 91.2 ve 88.62 elde ettik.

Ayrıca bu tez, VİT için etiketlenmiş ODTÜ derlemine dayanan ODTÜ-VİT derlemini de sunmaktadır. Lineer zincirli KRA modelini önceki veri setinde kullanılan aynı parametrelerle değerlendirdik. F1 ölçütü açısından kişi, yer, kurum, zamansal isimler ve genel olarak sırasıyla yüzde 73.26, 70.12, 63.83, 61.54 ve 69.14 elde ettik.

Anahtar kelimeler: Varlık İsmi Tanıma, Türkçe VİT derlemi, Dizi Sınıflandırıcı Modeller, Bilgi Çıkarımı

CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

CHAPTER ONE
INTRODUCTION

Textual Information Extraction (IE) is an important Natural Language Processing (NLP) task, which requires considerable attention due to the need to automatically extract semantic information from textual documents, such as Web news articles, and user generated content such as blogs, among others, that keep increasing in size daily. Current NLP techniques do not yet fully understand natural language articles. However, they may be useful for simpler tasks. Such as, one of them IE that is a language technology focusing on extracting structure from text contains some applications like text mining. The primary use of IE is to identify features that can be used by various applications such as search engines, machine translation, language processing, question answering, semantic tagging and automatic summarization. In IE, some of the text processing steps could be considered such as identifying and classifying noun phrases, titles or even bolded text. In each of these cases, a part of the text has been recognized as having some special property. The most important one is identifying and classifying noun phrases that is also called extracting the named entities.

The ability to extract the named entities from a natural language text is an important well-defined subtask in IE and is named as Named Entity Recognition (NER). We can roughly call the named entities as proper nouns or atomic elements in text. A named entity is often not simply a singular word, but is chunks of text that is used to refer something of interest in a particular application. The NER aims to identify and classify the named entities such as personal, geographic, institutional or other names (date, time, percentage and monetary expressions etc.) and none-of-the-above from natural language text. These words' type can be predefined or varied according to the problem worked on. NER is gaining importance with the rapid of increase, the Internet usage and generated content on micro blogs, social media sites, etc. With the recent advent of research topics like opinion mining, sentiment analysis, emotion extraction, social network analysis, event detection, topic extraction, other named entity types like product, emotion, brand and event names gained interest. For an ecommerce

application, for example, the recognition of product names and model numbers in web pages and reviews would be essential. In a pharmaceutical application, the recognition of drug names, dosages, and medical conditions may be important. In another words, NER research aims to convert informal text types like user-generated content to well-studied formal text types.

NER task has been introduced by the Defense Advanced Research Projects Agency (DARPA), and evaluated as understanding task in Message Understanding Conference (MUC). MUC was focusing on IE tasks where structured information of company activities and defense related activities are extracted from structured or unstructured text, such as newspaper articles. In IE task, people aim to recognize information unit like person, organization and location names. Identifying references to these entities in text was recognized as one of the important sub-tasks of IE. Several programs such MUC series, Conference on Natural Language Learning (CoNLL) and Automatic Content Extraction (ACE) programs had given rise to research on IE tasks and its subtask NER for vary languages such as English, Chinese, Japanese, and Arabic. The term "Named Entity" was coined for the Sixth MUC (MUC-6) (Grishman & Sundheim, 1996). In the MUC-6 and MUC-7 conference held in 1995 and 1998, NER systems achieved a significant success rate which is close to human performance as above 95% on the F1 measure. The early NER systems were designed for restricted domain and a particular language as English. These early systems were heuristic and were developed by extracting some features such as handcrafted rules. In modern systems, in addition to these features, dictionaries, machine learning based approaches have been continuing to work, and even deep learning methods are introduced. MUC and CoNLL conferences define three basic categories of named entities; these are 1- ENAMEX (person, location and organization names), 2- TIMEX (date and time entities) and 3- NUMEX (numerical expressions like money and percentages). In NER researches, the system's success is affected from language, text genre or data domain, and entity type factors.

1.1 Aim of Thesis

The objectives of this thesis are to provide an improvement on state-of-the-art NER on Turkish documents using a sequence classifier model on publically available dataset as well as it presents a new annotated dataset for NER studies in Turkish. We will discuss Hidden Markov Model (HMM) and Conditional Random Fields (CRFs) as sequence classifier models. The thesis also investigates to determine what the values of input parameters such as window size, output encoding format and features, which give the better results in a linear-chain CRF model and whether to design the HMM by representing each token with multiple features.

For Turkish, there are resource-constrained environment tools such as non-availability corpus, annotated corpus, name dictionaries, morphological analyzers, Part-of-Speech (POS) taggers etc., which are applied to the NER research as well. This thesis also aims to determine the extended feature set using less morphological analysis tools. Also, in order to make objective comparisons of different proposals, annotated datasets should ideally be publicly available. By presenting the new annotated dataset, we also contribute to the studies conducted in Turkish NER researches.

1.2 Thesis Organization

This thesis is organized in 7 chapters. In Chapter 2, we present an overview of NER and NER approaches with a literature survey on the subject. The sequence classifier models are one of the methods to solve NER problems. In Chapter 3, we give details about Hidden Markov Models (HMMs) and CRFs models, which are probabilistic approaches, have been experimented in this study. In Chapter 4, we explain the presented dataset, the annotation and segmentation of the data, the morphological analysis process, and the extraction of features in detail. Chapter 5 contains the details of methods that are used in this thesis and how we applied these methods on the datasets. The details of experimentations and the results are given in Chapter 6. Finally, a conclusion is made about the study in Chapter 7.

CHAPTER TWO
THE DEFINITIONS, APPROACHES AND RELATED WORKS IN NER

2.1 Information Extraction

IE is the task of extracting knowledge (particular types of entities, relations or events) from natural language dataset in various media types such as text, audio, video. The main goal of IE is finding valuable parts automatically. After IE process, the unstructured information embedded in texts converted structured data. This process is very important to solve most problems in hot research area such as question answering and summarization systems, information retrieval, machine translation, multimedia annotation, semantic Web search and bioinformatics. For example, in question answering systems, the key to a question processor is to identify the asking points (*wh*-questions such as who, when, where, what, etc.). These asking expressions correspond to a named entity. For biology text, there may be the predefined names like protein and DNA names that need to be extracted from raw documents.

IE is a language technology that focuses on extracting structure from text. IE is used in a number of applications, and particularly for text data mining. For search applications, the primary use of information extraction is to identify features that can be used by the search engine to improve ranking. Some people have speculated that information extraction techniques could eventually transform text search into a database problem by extracting all of the important information from text and storing it in structured form, but these techniques have a very long way from achieving that goal. The NER can serve various purposes, such as compact summaries of input texts or index terms in information retrieval systems, among others.

2.2 Named Entity Recognition

NER is one of the main tasks on IE. NER contains of two tasks, which the first one is the identification of some words like proper nouns in a natural language and in structured or in unstructured text. The second one is the classification of these names

into a set of predefined categories of interest such as person names, organizations, locations, and date and time expressions. These identified and extracted names are called "*Named Entity (NE)*". NER can also viewed as a classification task that assigns each NE (or entity mention) to its correct class, concept or entity. All of these NEs classes are known as "*Named Entity Hierarchy*".

NER approaches are usually adapted according to the following factors (Doğan Küçük, Arıcı & Küçük, 2017; Mosallam, Abi-Haidar & Ganascia, 2014):

- *Entity types* are also known as entity categories or concepts. The simplest NER techniques are used to extract coarse-defined NEs such as people, location, and organization. It is also common to extract fine-grained entities such as financial, bioinformatics or medicine etc.

- *Languages* are important factor that needs to be considered. English is one of the well-studied languages on NER domain. Turkish studies are relatively rare compared to other languages. So Turkish NER works are more challenging due to the lack of annotated data and knowledge bases. Doğan Küçük, Arıcı & Küçük (2017) described that there are only two studies that describe publicly available annotated corpora: an annotated tweet corpus (Dilek Küçük, Jacquet & Steinberger, 2014), an annotated news corpus (D Küçük, Küçük & Arıcı, 2016). In order to make objective comparisons of different proposals, annotated datasets should ideally be in public available.

- When designing a NER algorithm, *domains* such as business, medicine or information technology, and text genre such as journalistic, scientific or informal types have to be considered. Adapting NER techniques from one field to another has bad effects on accuracy as shown in Poibeau & Kosseim (2001). Various NER studies focus on news agencies due to their abundance and public availability.

2.3 NER Approaches

A lot of works in the NER researches have already been carried out using various approaches from handcrafted rule based systems to learning systems using different Machine Learning (ML) algorithms (Grishman, 2003). The method to be studied is selected according to the language, text genre or data domain and entity type factors. Some approaches are more successful to extract person names whereas some are good to extract location names so all the approaches are not good to detect all NE types.

2.3.1 Rule Based Approaches

In terms of world knowledge, the simplest and most relevant resource for NER task is a database of known names. Rule based approaches are relying on manually coded rules, lists or manually compiled corpora. Rule based approach is called the symbolic approach, can be classified as linguistic and list lookup approaches. In linguistic approach, the hand-crafted rules which are created by experts (linguists) in the specified language, should be satisfied by the particular word to belong to NE class or not. The rules are used to find patterns, which are based on various grammatical (e.g. part of speech), syntactic (e.g. word precedence) and orthographic features (e.g. capitalization) features. In list lookup approach, different lists can be formed containing NEs belonging to different NEs classes like person names, location names and organization names etc. The lookup operation or searching is executed on these lists for identification and classification of NEs. Main disadvantage of this approach is that the lists may not have all the NEs of a particular class.

Rule based approaches having better results for restricted domains, are capable of detecting complex entities that learning models have difficulty with. However, the rule based NE systems lack the ability of portability and robustness, and even if the data has a little change, the cost of maintenance of the rules is high. These type of approaches are often domain and language dependent and do not necessarily adapt well to new domains and languages and has very low performance. If rule based

system was designed for specific domain and then will be adapted to other domain, there will be required knowledge sources to be manually revised.

2.3.2 Machine Learning Based Approaches

In ML-based systems, the definition problem is considered as a classification problem, so a classification statistical model is preferred to solve this problem (Mansouri, Affendey & Mamat, 2008). In ML based approaches, the system looks for patterns and relationships in the text to generate a model using statistical models and machine learning algorithms. There are three types of ML models that are used for NER: supervised, semi-supervised and unsupervised.

2.3.2.1 Supervised Approaches

In supervised learning, system learns the classification with a given set of labeled examples, which is represented with the value of different features spaces. The training data are trained the system the right distinctions. So the supervised algorithms require a considerable amount of training data. But it is not feasible when considering their prevalent employment time-consuming. Also, the system cannot achieve a better result without a large amount of training data. So, the applicability of supervised methods is limited in the multilingual case. These approaches are also called the probabilistic approaches. In recent years, several statistical methods based on supervised learning methods were proposed: HMMs, Decision Trees (DTs), Maximum Entropy Models (MEMs), Support Vector Machines (SVMs), CRFs etc. A statistical entity recognizer uses a probabilistic model of the words in and around an entity. One of the major factors of success for such techniques is the availability of a huge annotated corpus in the target domain and language. We will give details about the HMMs and CRFs approaches into Chapter 3.

2.3.2.2 Semi-supervised Approaches

The applicability of supervised methods is limited in the multilingual case because of the cost of knowledge databases and manually annotated corpora. The semi-supervised ML is also called bootstrapping and includes little supervision like giving a set of seed to system for learning process. This bootstrapping procedure is to iteratively leverage relatively independent sources of information. Using some seed names for each class, the algorithm learns contextual patterns that are indicative for those classes. Then iteratively learns new class members and contextual and word-internal morphological clues. Through this cycle, probability distributions for class given token, prefix/suffix or context are incrementally refined.

2.3.2.3 Unsupervised Approaches

The aim is to recognize NEs in a given document without prior training (supervised learning) or manually constructed dictionaries. Also we can divide unsupervised techniques into two categories: clustering and knowledge based resources. Due to the vast amount of available knowledge sources, this approach became popular. Unsupervised NE extraction systems do not need human intervention. The learning process occurs without any feedback in these approaches. The performance of systems using unsupervised ML approaches is low compared to supervised ML approaches. Unlike the rule-based methods, the main advantage of unsupervised learning approach is that they can be easily port to different domains or languages.

2.3.3 Hybrid Based Approaches

In Hybrid NER system, the approach is to combine rule based and machine learning based methods, and produce new methods using strongest points from each method. For example, in a system that is used a hybrid based approach, there can be applied a combination of some approaches such as HMM or MEM and handcrafted grammatical rules. In this approach, although the better results can be obtained, the disadvantages

of used approaches continue in the same for example the weakness of handcraft rule continues when the domain of data changes.

2.4 Related Works

State-of-the-art NER systems have been studied for several languages. The method has been preferred in NER studies vary according to the characteristics of dataset language. In languages such as English, there are very few possible word forms with a given root word. But, in morphologically rich languages, which have very complex morphological structure such as Turkic languages (Turkish, Azerbaijani, Gagauz and Oghuz etc.), Finnish, Czech, Korean, Hungarian and many others, there are thousands of words from a given root. A sequence of inflectional and derivational morphemes can be added to a word in Turkish and produce different surface forms. So millions of different surface forms can be derived from a nominal or verbal stem (Hankamer, 1989). After adding morphemes to a word, generated long length words can convey the equivalent meaning of a phrase. In addition, a Turkish word can contain different variants of meaning in the dictionary. These surface forms, which are generated from the same stem, cause data sparseness problems in model training especially ML based approaches. So, in morphologically rich languages, the success rate of NER systems can remain still behind the reported accuracies for other languages such as English. The reported results in similar studies in languages such as English are around %95. The morphological information is used in studies to overcome data sparseness problem. However, the usage of morphological information for the NER task is still an open research issue in these languages.

The NER task is significant research topic for languages other than Turkish such as English, Spanish, Chinese and Japanese. The NER researches on these languages surveyed in these studies Cunningham (2005), Grishman (2003), Nadeau & Sekine (2007), Turmo, Ageno & Català (2006).

Turkish uses a Latin alphabet consisting of 29 letters, of which 21 are consonants and eight are vowels. It is a member of the Oghuz group of the Turkic languages,

which belongs to the Altaic branch of Ural-Altaic language family. Turkish is a highly agglutinative language with free-word order (Oflazer, 1994) and it has very productive inflectional and derivational processes and so it is a complex language as morphologically. In agglutinative languages the meaning of the word can be changed when it takes each affix. The NER studies on Turkish texts are relatively rare compared to other languages such as English, Chinese and Spanish. When compared to studies in languages such as English, stemming and tokenization methods are not performed well for Turkish. When a stemming method is applied, it may cause to lose important information. We intensively analyzed NER studies in literature in terms of dataset used, annotation phase, methods or approach, features, scope end performance result. Table 2.1. gives a summary of them where details are given in Appendix 3.

Table 2.1 Comparison of Turkish NER systems

Author	Dataset	NE Type	Approach	Eval.	F-Measure
Cucerzan & Yarowsky (1999)	5207 words in text	PERSON LOCATION	EM-style bootstrapping	MUC	53.04
Tür, Hakkani-Tür & Oflazer (2003)	Milliyet newspaper articles	ENAMEX	HMM	MUC	91.56 (PER:92.63;LOC:95.33; ORG:89.77)
Bayraktar & Temizel (2008)	Economy Corpus and METU Corpus	PERSON	Local grammar based	MUC	81.97
Küçük & Yazıcı (2009)	METUCorpus(I) child stories (II) historical texts (III)	ENAMEX, NUMEX, TIMEX	Rule based	MUC	(I) 78.7; (II) 69.3; (III) 55.3
Adali, Sonmez & Gokturk (2009)	500 (I)/1000 (II)/ 2000 (III) financial documents	Financial	Rule based + dictionary based	MUC	(I) 97; (II) 98; (III) 99
Dalkılıç, Gelişli & Diri (2010)	10 documents in politics (I) economy (II) health (III)	ENAMEX	Rule based	MUC	(I) PER:0.78; LOC:0.84; ORG:0.73 (II) PER:0.88; LOC:0.90; ORG:0.82 (III) PER:0.79; LOC:0.90; ORG:0.85
Dilek Küçük & Yazici (2010)	METUCorpus (I) financialnews(II) child stories (III) historical texts (IV)	ENAMEX, NUMEX, TIMEX	Hybrid	MUC	(I) 85.95; (II) 74.23; (III) 85.06; (IV) 66.96
Ozkaya & Diri (2011)	150 e-mails in academic (I), business (II), personal (III)	ENAMEX	CRF (CRF by Sarawagi)	MUC	(I) PER:0.94 LOC:0.68 ORG:0.83 (II) PER:0.97 LOC:0.79 ORG:0.84 (III) PER:0.92 LOC:0.70 ORG:0.90

Table 2.1 continues

Tatar & Cicekli (2011)	TurkIE	ENAMEX, TIMEX	Automatic rule learning	MUC	91.08 (PER:94.33 LOC:90.03 ORG:87.68 DATE:96.50 TIME 92:05)
Yeniterzi (2011)	Milliyet newspaper articles	ENAMEX	CRF (CRF++)	CoNLL	88.94 (PER:89.32 ORG:83.50 LOC:92.15)
Dilek Küçük & Yazici (2012)	METUCorpus (I) financialnews(II) child stories (III) historical texts (IV)	ENAMEX, NUMEX, TIMEX	Hybrid	OTHER	(I) 90.13; (II) 76.80; (III) 92.47; (IV) 80.66
Seker & Eryigit (2012)	Milliyet newspaper articles	ENAMEX	CRF (CRF++, [-3,+3], IOB2)	MUC	94.59 (PER:94.81 ORG:91.09 LOC:93.35)
				CoNLL	91.94 (PER:92.94 ORG:88.77 LOC:92.93)
Kazkilinc & Adali (2013)	75 documents for different categories	SUBJECT,VERB,LOCATION,TIME	CRF	MUC	SUB:72.00 VERB:68.4 LOC:69.5 TIME:55.00
Çelikkaya, Torunoğlu & Eryiğit (2013)	Milliyet newspaper articles (I), Forum (II), Speech-to-Text (III), Twitter (IV)	ENAMEX	CRF (CRF++,[-3,+3])	CoNLL	(I) 91.64; (II) 19.28; (III) 50.84; (IV) 5.62
Yavuz, Küçük & Yazıcı (2013)	METUCorpus	ENAMEX,TIMEX,NUMEX	Bayesian Learning (I) and rule based (hybrid) (II)	OTHER	(I) 87.99 (PER:90.30; LOC:87.06; ORG:88.03; DATE:83.13; TIME:80.28; MON:84.73; PERC:92.01) (II) 90.05; 91.44
Demir & Ozgur (2014)	Milliyet newspaper articles and different source news	ENAMEX	semi-supervised (neuralNetworks,BIEWO)	CoNLL	91.85 (PER:94.69;ORG:85.78;LOC:92.40)
Eken & Tantug (2015)	Milliyet newspaper articles,Tweet dataset[1], Tweet dataset[2]	ENAMEX, NUMEX, TIMEX	CRF++ (IOB2)	CoNLL	63.77 (on Tweet dataset1)
Emekligil, Arslan & Agin (2016)	Bank Customer Transaction Orders	ACC,AMOUNT,CLIENT,CURR,EXPL,IBAN,ORG	CRF (MALLET, [-3,+3])	CoNLL	72.77
Seker & Eryigit (2017)	Milliyet newspaper articles (I), Forum (II), Twitter (III)	ENAMEX, NUMEX, TIMEX	CRF (CRF++, [-3,+3], IOB2)	CoNLL	(I) 92.34 (PER:91.47 ORG:94.34 LOC:89.88 DATE:89.25 TIME:91.89 MON:100 PERC:98.41)
					(II) 64.96 (PER:67.22 ORG:77.17 LOC:53.87 DATE:70.31 TIME:80.0 MON:27.45 PERC:50.0)
					(III) 67.96

The first study to be considered is the one by Cucerzan & Yarowsky (1999) aimed to using minimal information about the source language and building a maximally language-independent system for NER. They presented a language-independent Expectation Maximization (EM)-style bootstrapping algorithm based on iterative learning and tested on Turkish texts in addition to other texts in Romanian, English, Greek, and Hindi. In this system, training is applied on a small training data and learns from word internal and contextual information of entities, which are unannotated text with bootstrapping algorithm. They used some common prefixes and suffixes such as *"Mari-", "-escu", "-ovic", "-wski"* and *"-ivic"* for certain types of named entities which refer to the morphological information. They traced some contextual patterns such as *"Mr.", "in", "mayor of"* in left context because it also gives clearly crucial to NER. They used word separators (such as spaces or punctuation) to tokenize the text. Therefore, this tokenization method presents the advantage keeping tokens and types. They considered character based tries, in which each node contains a probability distribution. In each node, two distributions are considered, one for tokens and one for types. They used short name lists which has 40-100 example words as seed data for each entity class. They achieved levels of precision as 60.38%, recall as 47.29% and F-Measure as 53.04% for NER from Turkish texts.

The first NER study focus to Turkish is Tür, Hakkani-Tür & Oflazer (2003) which presented HMMs based NER system and used the following four models: lexical, contextual, morphological and name tag. Name tag model is type of named entity. The other models were used for features. In lexical model, the lexical information is captured using only word tokens. In contextual model, the surrounding context of the word tokens is used. Case and name tag information is captured in morphological model. They aim to extract ENAMEX types from Turkish texts. They used lexical, morphological and contextual features of the words to generate an HMM based model. They used dataset consists of nearly 500K words collected from newspaper articles. They reported the results using MUC evaluation metrics and evaluated using F-Measure. In their system, training data has ~493K words of newspaper articles and ~38K named entities. In test data, they have 28K tokens and ~2.2K named entities. Accuracy of their system has reached 91.56% when they have combined all models.

In a study, they evaluated local grammar based approach on Turkish financial texts (Bayraktar & Temizel, 2008). They aim to find the significant reporting verbs (RVs) in Turkish texts and evaluate local grammar (LG) approach for Turkish texts. LG were employed for various purposes and one of them is information extraction. LG aims to recognize the behavior of words used in a specialist context and find the usage in the sentences and infer patterns in usage. They used Economy Corpus (EC2000) and METU Turkish Corpus. The EC2000 was taken from Anadolu Agency and contains ~9.1M words. They used to comprise 200 news having 41,322 tokens as a testing dataset. They wanted to extract named entities including the names of people. They based their approach on the bootstrap method. For extract person names, they applied some prerequisites processes (preprocessing the datasets, performing word frequency analysis, extracting Turkish reporting verbs, testing statistical significance of Turkish reporting verbs) and the following four steps to extracting person names: concordance analysis, collocation analysis, pattern generation and extracting person names. F-Measure value by using the LG approach and capitalization rule (CR) together is better than only CR feature and they achieved F-Measure results as 81.97%.

In other study, Küçük & Yazıcı (2009) wanted to extract named entities including the names of people, locations, organizations, time/date, and money/percentage expressions following the specifications of the MUC series. They worked on METU Turkish Corpus (Say, Zeyrek, Oflazer & Özge, 2002), child stories and historical texts. They presented rule based NER system, which employs a set of lexical resources and pattern bases. They did not make use of capitalization and punctuation clues. They used some lexical sources: person names (it has 8,300 names), list of well-known people, location and organization names, and patterns (succeeding words, preceding tokens of keywords) which is used for expressing location and organization names, and temporal and numeric expressions. They annotated 10 news articles from METU Turkish Corpus with MUC format using own annotation tool. Their F-Measure results are 78.7% on news articles, 69.3% on child stories and 55.3% on historical texts. The aforementioned rule based recognizer has also be successfully integrated into automatic and semi-automatic semantic video annotation systems for Turkish news videos (Dilek Küçük & Yazıcı, 2011, 2013). In these systems, named entities are

automatically extracted from video texts to be used as semantic annotations. In their other study, they presented a hybrid named entity recognizer as a preliminary version (Dilek Küçük & Yazici, 2010). They used METU Turkish Corpus (101,700 words-11,206 NEs), financial news texts (84,300 words-5,635 NEs, from Anadolu Agency and only person and organization names are tagged), child stories (19,000 words-1,084 NEs) and historical texts (20,100 words-1,173 NEs). The system learns from annotated data through rote learning and enriches its knowledge sources accordingly. They used rote learner as learning approach which use two statistical features: the number of occurrences of the entity and the number of occurrences which happen to be annotated. They achieved the better results rather than their previous work on the same corpora, the F-Measure results are 85.95% on news, 85.06% on child stories, 66.96% on historical texts and additionally 74.23% on financial news texts which were collected from news agency. In their main work is based on a hybrid NE recognizer (Dilek Küçük & Yazici, 2012), they achieved as the results that 90.13% on news, 76.80% on financial news, 92.47% on child stories and 80.66% on historical text when capitalization feature is turned on.

Adali, Sonmez & Gokturk (2009) designed an integrated information extraction architecture for processing business documents in Turkish which is applied on restricted domains such as financial documents. The architecture has document layout analysis, ontology based information extraction and morphological analyzer for Turkish. In document processing stage, they tokenized the input document and stored them with row and column index in array structure to keep track of documents boundaries and templates. In NER, they defined some rule such as date format and used some lexicon tables such as customer names, banks, city names, and branch names. They applied an incremental lexicon search for tokens on lexicon databases. In document modeling and layout analysis stages, a document is classified by similarity of expressions, style and page layout. Document layout analysis begins by grouping the primitive elements of the input document to higher-level objects, such as paragraphs and headings. The template matching technique is used to match blocks of a predefined document block templates and an input document. The proposed architecture aims to create a minimum-definition event extraction mechanism using

field concepts and reuse these concepts to create a portable and scalable IE system. Ontological information and morphological analysis results in the proposed architecture are used together for domain filtering. In their experimentations, they achieved as F-Measure 97% for 500, 98% for 1000 and 99% for 2000 financial documents with ontology based IE and document layout analysis. System extracts information such as date, time, fax number, currency type, customer name, account number and money amount related to a specific scenario.

The other study focuses on extract named entities including the names of people, locations, and organizations in topic independent Turkish documents that contained three categories that has 10 text files for each category, were political, economy, and health (Dalkılıç, Gelişli & Diri, 2010). For location and organization names, some clues were used: suffixes, next context and small size databases which are too small for to be called gazetteers. In addition to extracting organization names, name abbreviation list was used. The dictionary of person titles and verbs was used extracting person names. They presented rule based NER system. They make use of capitalization and punctuation clues. They achieved the better F-Measure result in economy documents is 88% in person, in economy and health documents 90% in location, and in health documents 85% in organization. Their NER system was used in this information retrieval system finding people's expressions (Ozdemir & Diri, 2011). Personal identification is important for retrieving relevant pages in this system. This system aims to search "*What person X told about the topic Y?*" on the internet or in the database of the system.

In a study, CRFs has been used to extract person, location and organization names from informal texts (Ozkaya & Diri, 2011). They used CRF package (Sarawagi, 2005) for sequential labeling. Some features and rules were extracted from each token: word, length of word, frequency of word, is abbreviation or not, position in sentence, type (noun, adjective, adverb etc.), case of initial letter, has punctuation or not, exist in title or content etc. They used abbreviation, title of person, special words gazetteers. The dataset consists of 150 e-mails collected in equivalent count from academic, business and personal e-mails. The better results were obtained in business e-mails and F-

Measure was calculated 97% in person names and 79% in location names. In organization names, the better F-Measure rate was 90% in personal e-mails.

In another study, they used different features of the input text to identify the named entities (Tatar & Cicekli, 2011). They described it as automatic rule learning method. They conducted on the TurkIE dataset consists of a corpus of articles which different Turkish newspapers. The corpus contains 54,518 tokens, 3,524 sentences and 545 topics. They followed the MUC-7 NE task definition as a guideline for annotation and manually tagged 355 articles on terrorism using IE-tagging tool. 5,672 NEs were tagged in five types: person, location, organization, date and time. They used the following categories of features: lexical, morphological, contextual and orthographic. In lexical features, they build two level gazetteer hierarchy using gazetteer information. They followed the standard tokenization method using white-space characters and punctuation marks without apostrophe. They used morphological analyzer and disambiguator in morphological feature. Contextual information was captured from the surrounding text in contextual features. They selected four primitive features as orthographic features: capitalization, length class, length of the token and token type class. They used two type of sequence patterns presenting the rules: similarity (SIM) and optional (OPT). Each pattern has composed of features value. In SIM sequence, each feature value exactly matches one token and for features existed in OPT sequence meets the token's constraints or not. Automatic rule learning is one of the critical functions in the system. They take two examples in the rule representation and return a generalized rule in the generalized function. They calculated the confidence factor of a rule for the success rate and define the confidence factor threshold. They applied 10-fold cross validation on the dataset and their method achieved an average F-score of 91.08%. In their experiments, the biggest lost occurs when morphological features were not used.

In a study, Yeniterzi (2011) analyzed the effect of morphological features like root, Part-of-Speech (POS) tag, proper noun and case in a NER system for Turkish. They applied to the first CRF based method and used training set of the newspaper articles data (Tür et al., 2003). They split the dataset into two clusters, with 90% of the train

data for training and left the remaining for testing. They aimed to extract three types of named entities: person, organization and location. In the tokenization step, in addition to the word-level based tokenization method which used in previous studies, they used a new tokenization method which represents each root and morphological features as separate tokens. They applied the Turkish morphological analyzer tool (Oflazer, 1994) and then used morphological disambiguator (Sak et al., 2008) to choose the correct morphological analysis of the word depending on the context. In their experiments, they used CRF++ toolkit (Kudo, 2005) is an open source CRF sequence labeling toolkit and CoNLL metrics to evaluate. According to results obtained, morpheme-level model achieved better than word-level model in person and location entities and in terms of recall. Even though word-level model is better than morpheme-level in organization entities. They obtained the best results in precision, recall and F-Measure values as 91.81%, 86.87% and 88.94% in overall, respectively.

Seker & Eryigit (2012) used the rich morphological features as features in CRFs with some basic and generative gazetteers. They represented each word as a token and considered all punctuation marks as a token. In proper nouns, they partitioned such them into two tokens: the tokens before and after the apostrophe. They separated sentences from each other with an empty line. They used a two-level morphological analyzer (Oflazer, 1994) and give its output to morphological disambiguator (Sak et al., 2008) to get the most probable analysis in the given context. They used the stem of the word and the morphological features as features for CRF model. In addition to, they used two kinds of gazetteers: base and generator. Base gazetteers are large gazetteers with thousands of tokens, which are collected from different sources and include words with high probability of occurring in a named entity. They used these gazetteers for first names (44,048 tokens), surnames (138,844 tokens) and location names (33,551 tokens). Generator gazetteers are smaller than base gazetteers. They contain stem of some basic words and when they are used with some words will be named entity. The generator gazetteers were used for location, organization and person names. They used the labels for person, location, organization names and other. In other label, the words are not named entity. They used CRF++ tool (Sarawagi, 2005) which is open source implementation of CRFs. They compared labelling the output

with two different type of tags: raw and IOB2. They used the data (Tür et al., 2003) in their experiments. As morphological features, they used some information: stem of the word, POS tag, noun case, proper noun or not, all inflectional features. In lexical features, they used case feature and start of the sentence information as feature. They used the features in CRF model within a window of [-3, +3]. The F-Measure results were obtained as 94.59% and 91.94% according MUC and CoNLL evaluation metrics, respectively. In another study which is a continuation of this study, they extended their model for different data types and they conducted model on user generated content (Seker & Eryigit, 2017). In addition to other work, they aim to extract and NUMEX NE types. They expanded the gazetteers size and added new gazetteers for different NE types such as months, currency units. They used extra features for TIMEX and NUMEX types such as numeric value, contains or not percentage sign, o'clock term, column indicator and exists or not in month and currency gazetteers. For user-generated content (Twitter), they applied some normalization steps on data. They conducted all experiments on three-test dataset (WFS7, TDS, IWT) using one training set (TRAIN7). In section 4.1, we will give details about these datasets.

In another study, they worked on labeling the main subject, predicate, location and date in Turkish news stories (Kazkilinc & Adali, 2013). Finding important phrases from a document facilitates extracting expected information. So they aim to extract subject, verb, location and date expressions from text. The documents are about Turkey, World, political, economic, science, technology and sports in dataset. Each category has 75 documents, which collected from the Internet news sources. They used morphological analyzer (Oflazer, 1994) and disambiguator (Sak et al., 2008) for extracting morphological features. After they applied dependency parser (Eryiğit, 2007) on morphological analysis to find their tasks within the sentence. In CRF system, they defined some feature categories: rule based, morphological, syntactic and structured. In rule based features, some information was used: case of initial letter, position in sentence, contains or not prop tag in morphological analysis, special suffixes in boundary of word, contains or not comma and apostrophe. Morphological and syntactic features were obtained from morphological analysis and dependency parser. Order of text and sentence, frequency of token and first observation position

were used in structured features. Their F-Measure results were 72% in subject tag, 68.4% in verb tag, 69.5% in location tag and 55% in date tag.

Çelikkaya, Torunoğlu & Eryiğit (2013) used linear chain CRFs within the window of [-3,+3] as the machine learning algorithm. They used the CRF++ library (Kudo, 2005). on different domains' data collected from 3 different domains: social media content (Twitter), spoken language data (a Speech-to-Text Interface) and a hardware forum website. They used the new three datasets as the test data and consist of approximately 55K, 1.5K and 55K tokens, respectively. They annotated manually according to the NER guidelines of MUC6 ("MUC-6", 1996). For training data, they used the data which was collected from Turkish Newspaper site consists of ~500K tokens. They divided into two parts as training (450K) and test set (50K). Firstly, they tokenized the data and prepared training and test instances by the use of the morphological and predefined features. They used the following items as morphological features: stem of the token, the main POS-tags, the case marker (for nouns) and the availability of the proper noun tag. The other features of each token are: a surface form of the token, its upper/lower case status and some binary features stating the sentence beginning or not, the existence in the prepared gazetteers. They performed some process stages to avoid the observed problems specific to data. For example, in the real data, some spelling errors are occurred and capitalization or punctuation rules is not properly used. Before other stages, they created a text normalizer tool which works on the following cases: slang words, repeated characters, emo style writings, hash tags, mentions, vocatives, smiley icons, capitalization. They prepared 6 gazetteers: first names (~45K), surnames (~140K), and location names (~34K) are base gazetteers; location, organization and person are generator gazetteers. They extended gazetteers while adapting to the real data. They have developed three different feature models which have some selected features and used CoNLL metric on the evaluation (Nadeau & Sekine, 2007). They only evaluated for ENAMEX types and experimented these models with and without normalization and added capitalization to them. For Twitter and speech data, the best results obtained with all features and normalization without capitalization 15.27% and 50.84%; for news media

and forum data, the best results obtained without normalization and some features (lower/upper case, first word) 91.64% and 5.62%.

Yavuz, Küçük & Yazıcı (2013) presented a Bayesian learning approach which is trained on a considerably limited training set and comprise the Bayesian learning with a previously their rule based NER. They modified the Bayesian learning approach and used some parameters and features that are extracted from tokens into it. These parameters correspond to the number of tokens to be considered before and after named entity instances in the training set and case sensitivity. The features are case, the length of token, alphanumeric, nymble and lexical resource features. They used a Bayesian with their previous rule based approach into two phases: training and estimation. In estimation phase, they used the rule-based recognizer's results into Bayesian as a distinct feature. In training phase, before carrying out the Bayesian estimations on the test data, the rule based NER run on the test data and the resulting is utilized as an additional training data to update the probabilities. They have used one of the datasets which was annotated into their previous study (Dilek Küçük & Yazici, 2012). This news dataset comprises 50 news articles (101,700 words) from METU Turkish Corpus. The annotated dataset contains 3,280 person names, 2,470 location names, 3,124 organization names, 1,413 date/time and 919 money/percent expressions. They have evaluated using precision, recall and F-Measure with ten-fold cross validation. In test dataset, it has 228 NE (102 persons, 42 locations, 71 organizations, 11 date, 1 money, 1 percent). They obtained the F-measure results for person names 90.30%, location %87.06, organization 88.03%, date %83.13, time 80.28%, money 84.73%, percent 92.01 and overall 87.99 using Bayesian learning and for overall 90.05% with training-base hybrid and 91.44% estimation based hybrid recognizer.

Demir & Ozgur (2014) used the neural network based semi-supervised learning approach for NER in morphologically rich languages, Turkish and Czech. They provided to improve F-score obtained for Turkish NER system by 2.26% and for Czech NER system by 1.53% without using dictionaries. In their system, any language dependent features have not been used and their system consists of two stages:

unsupervised and supervised. In the unsupervised stage, they used new model architectures studied into Mikolov, Chen, Corrado & Dean (2013) for learning distributed representations of words. It makes use of a huge amount of unlabeled data. They collected data from Turkish news sites and tokenized and then lowercased them in this stage. The data has 63.72M sentences, 1.02B words, and 1.36M unique words. In the supervised stage, in addition to presented feature vectors, they used other additional language independent features such as capitalization patterns, previous tag prediction, etc. In this stage, they used the labeled dataset prepared by (Tür et al., 2003) which is the most commonly used one for evaluating Turkish NER system. In their system, they reported results with respect to the CoNLL metric. They achieved as F-Measures 94.69% into success rate for person names, 85.78% for organization names, 92.40% for location names, and 91.85% in the overall.

For improving NER on informal text domain such as social media texts like tweets, they aimed to increase the performance of Turkish NER in Turkish tweets (Eken & Tantug, 2015). They used CRF tool (Kudo, 2005) to build their NER model and two main dataset from two different domain: news (Tür et al., 2003) as a formal data and tweets as informal data. Tweet dataset consisting two parts, first part consists 9K tweets and labelled by them for this study. The second part is used into this study (Çelikkaya et al., 2013) and consists of nearly 5K tweets. They used Oflazer's tool (Oflazer, 1994) for morphological analyses and Sak's tool (Sak et al., 2008) for morphological disambiguation. They used the first and last four characters of tokens as a feature into CRF model. They inferred that there is no need to use morphological analyses tools. They applied the Levenshtein distance algorithm to calculate distance between two strings: token and words are existed into gazetteers. They evaluated their results CoNLL metric and presented entities IOB2 representation style. They reported three main results after their experimentations. They used morphological features, letter case features, start of sentence features and gazetteers to build model into first model. In another model, they build it without morphological features and took the first and last four characters as an alternative stem. They used to contain or not apostrophe, case state, position information in a sentence, gazetteers lookup and distance based matching feature in second model. They achieved the results as F-

Measure like that 90.47% on news test set, 41.79% on tweets first part test set, 25.62% tweets second part test set, and 90.23% on news, 46.57% on tweets first part and 28.53% on tweets second part into overall, respectively. In last one, they used tweets as a training and test data and reported 63.77% as a result.

In another study, Emekligil, Arslan & Agin (2016) used CRFs for extracting required named entities (client name, organization name, bank account number, IBAN number, amount, currency and explanation) from customer transactions. These transactions such as money transfer, salary payment, tax payments etc. are received in different formats via fax channel at banks operation center. This information is referred as named entities are used in a maker-checker basis. In some bank systems, these entities are extracted manually by operators. They presented a fast and robust information extraction system to extracted necessary entities from optical character recognized (OCR) Turkish customer order documents. They used OCR tool to convert fax documents to text. The fax documents have no common format as provided from the bank. They annotated 365 real transaction order documents. They take the first five characters of words for do stemming. They used person name gazetteer which has ~13K Turkish person first names provided by the bank. They changed digit letters to '*d*', punctuation to '*p*', lowercase to '*s*' and uppercase to '*S*' in each word. They used it as letter shaped features and some Boolean features such as begin parenthesis and ends with a colon. They used MALLET Framework (McCallum, 2002) on their experiments. Initially, they compared HMM and CRFs and they decided CRF in further experiments when they were taken better results. When they used [-3, +3] window for each token, overall CRF performance is increased by 21%. Their F1 scores of method with all features on 5-fold cross validation and test set are 78.89% and 72.77% for average in CoNLL metric.

CHAPTER THREE
SEQUENCE CLASSIFIER MODELS

3.1 Overview

Most things in real life are a mixture of random and deterministic relationships. For example, weather patterns are partly random and can be partially estimated. When something is part random and part partial deterministic, it is called a statistical relationship and a probabilistic relationship, respectively.

A *stochastic model* represents a situation in which uncertainty exists. In other words, it is a model for a process with some kind of randomness. In reality, uncertainty is a part of everyday life, so a stochastic model could literally represent anything. Stochastic models will likely produce different results each time the model is run. Probabilities are assigned to events within the model and these probabilities can be used to make estimates or provide other information about the process in all stochastic models (Andale, 2016a). The opposite of a stochastic model is a *deterministic model* that predicts results with 100% accuracy. Deterministic models always have a set of equations that define the system inputs and outputs exactly (Andale, 2016b). For example, the conversion between Celsius and Kelvin is deterministic because the formula is not random - it is always a definite formula to give you the right answer if you perform the calculations correctly.

The "probability" part of the class includes calculating probabilities for events happening (Kay, 2006). Probability as defined by Webster's dictionary is "the chance that a given event will occur." Examples, which we are familiar with, are the probability that it will rain the next day or the probability that you will win the lottery. In probability, a result of a random experiment is called "*outcome*"; the set of all possible outcomes is called "*sample space*" (is denoted by S) and a subset of the sample space is called "*event*" (is denoted by E). A "*random experiment*" is an experiment in which the result changes unpredictably when the experiment is repeated under the same conditions (Leon-Garcia, 2011). A random experiment is indicated by

specifying an experimental procedure and one or more measurement or observation sets. The following probabilities are used for the probability of the occurrences according to event (Koller & Friedman, 2009; Leon-Garcia, 2011).

- "*Joint probability*" is the probability that two events will occur simultaneously. It is the probability of the intersection of two or more events. For example, the probability that a card is a four and red is 2/52. (There are two red fours in a deck of 52, the 4 of hearts and the 4 of diamonds.) It is shown as $P(A\ and\ B) = P(A \cap B) = P(A, B)$

- "*Marginal probability*" is the probability of the occurrence of the single event, it may be thought of as an unconditional probability. It is not conditioned on another event. For example, the probability that a card drawn is red is 26/52. It is shown as $P(A)$.

- "*Conditional probability*" (3.1) is the probability of event A occurring, given that event B occurs. Example: given that you drew a red card, what's the probability that it's a four is 2/26. It is formulated as for $P(B) > 0$:

$$P(A|B) = P_B(A) = \frac{P(A \cap B)}{P(B)} \tag{3.1}$$

If knowledge of the occurrence of an event B does not alter the probability of some other event A, then it would be natural to say that event A is "*independent*" of B. An independent event has no connection to another event's chances of happening or not happening. The event has no effect on the probability of another event occurring. If $P(A \cap B) = P[A]P[B]$ then implies both $P(A|B) = P(A)$ and $P(B|A) = P(B)$, events A and B independent.

Those types of probabilities are defined above use to design some models such as "*discriminative*" and "*generative*" models.

- In discriminative models, it learns the boundary between classes and more powerful with lots of examples. Discriminative models are designed only supervised tasks so need to use labeled data only. It describes how to take a feature vector x and assign it a label y.

- In generative models, it models the distribution of individual classes. This model design probabilistic model of each class and use unlabeled data. It describes how a label vector y can probabilistically generate a feature vector x.

Figure 3.1. shows some of the well-known probabilistic models such as Naïve Bayes (NB), HMM, MEM and CRF, and their differences.

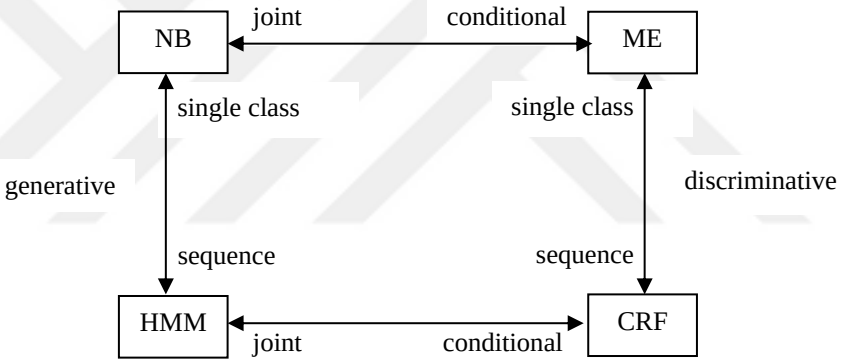

Figure 3.1 Overview of probabilistic models

The NB Model is an approach to classify individual variables based on several property values. In this model, the input values are assumed to be conditionally independent. It is a so called generative approach, which modeled the joint probability $\mathrm{p}(y, \vec{x})$ of the input values $\vec{x}$ and the class variable y. In classification, predicting a single discrete class variable y given a vector of features $x = (x_1, x_2, \dots, x_K)$. The HMM is an extension of the NB Model for sequentially structured data representing dependencies of the variables $\vec{x}$ and $\vec{y}$ as a joint probability distribution. Modeling joint probabilities has disadvantages due to computational complexity. The MEM is based on modeling the conditional probability $\mathrm{p}(y|x)$. It is an approach to classify a single class variable in dependence of several feature values. While a HMM is a

sequential extension to the NB Model, CRFs can be understood as a sequential extension to the MEM. Both MEMs and CRFs are known as discriminative approaches (Klinger & Tomanek, 2007).

A "*sequence model*" or "*sequence classifier*" is a model whose job is to assign a label or class to each unit in a sequence, thus mapping a sequence of observations to a sequence of labels. The standard algorithm for NER is as a word-by-word sequence labeling task, in which the assigned tags capture both the boundary and the type (Jurafsky & Martin, 2014). So in this thesis, we compared HMM and CRF on our experimentations.

3.2 Markov Chains

The Markov Property is a characteristic of a stochastic process where the conditional probability distribution of a future state depends on the current state and not on its past states. In this case, the transition between the states occurs at a discrete time, and the *Markov Property* is known as the *discrete Markov chain*.

A *Markov Chain*, sometimes called the *observed Markov model* is extension of the finite automata is shown in Figure 3.2 (Alpaydin, 2009; Jurafsky & Martin, 2014). A weighted finite automaton is defined by a set of states and a set of transitions between states, with each arc associated with a weight. The idea behind Markov chains is usually summarized as follows: "conditioned on the current state, the past and the future states are independent."

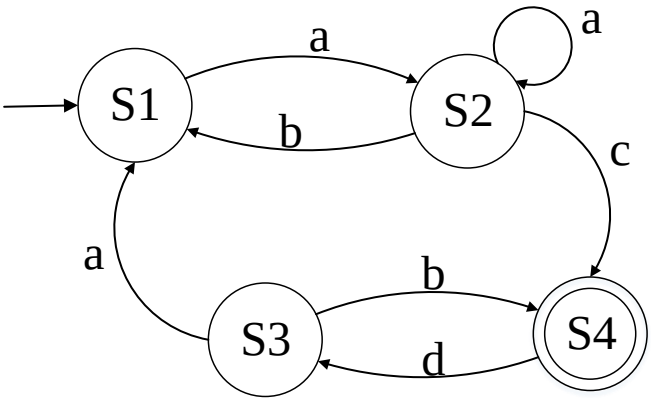

Figure 3.2 Example of a finite automaton

A Markov chain is a special case of a weighted automaton in which weights are probabilities (the probabilities on all arcs leaving a node must sum to 1) and in which the input sequence uniquely determines which states the automaton will go through. Because it can't represent inherently ambiguous problems, a Markov chain is only useful for assigning probabilities to unambiguous sequences.

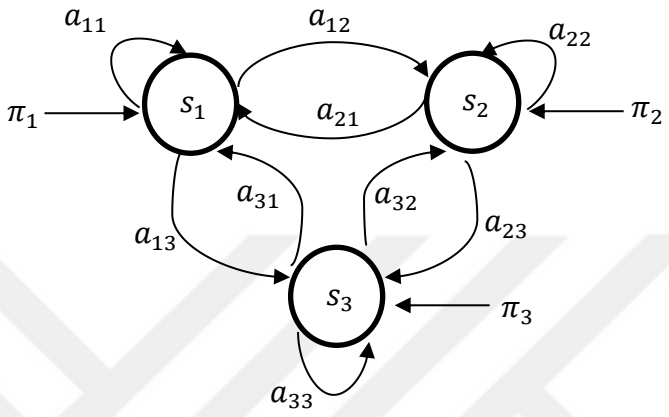

Figure 3.3 Example of a Markov model with three states

Figure 3.3 shows that we represent the states (including start q_0 and end q_F states) as nodes in the graph, and the transitions as edges between nodes. This is a stochastic automaton where π_i is the probability that the system starts in state s_i and a_{ij} is the probability that the system moves from s_i to state s_j. We describe a Markov chain is described by the following components, consider a system that at any time is in one of a set of *N* distinct states:

- *The state space (set of possible states):*

$$S = \{s_1, s_2, \ldots, s_N\} \tag{3.2}$$

- *The prior (initial) probabilities*

$$\pi_i \equiv P(q_1 = s_i) = \frac{\#\{sequences\ starting\ with\ s_i\}}{\#\{sequences\}} \tag{3.3}$$

The state at time/position t is denoted as $q_t, t = 1,2, \ldots$ so, for example, $q_t = s_i$ means that at time/position t, the system is in state s_i (3.3). Probabilities of s_i being the first state of a state sequence. Collected in $\Pi = [\pi_i]$ is a vector of N elements that sum to 1 (3.4). (The prior probabilities are often assumed equi-probable, $\pi_i = 1/N_s$)

$$\sum_{i=1}^{N} \pi_i = 1 \tag{3.4}$$

- *The transition probabilities*

For the special case of a first-order Markov model, the state at time $t + 1$ depends only on state at time t, regardless of the states in the previous times:

$$P\left(q_{t+1} = s_j \middle| \, q_t = s_i, q_{t-1} = s_k, \ldots\right)$$

This is Markov assumption and corresponds to saying that, given the present state q_t, the future q_{t+1} is independent of the past q_{t-1}.

$$P\left(q_{t+1} = s_j \middle| \, q_t = s_i, q_{t-1} = s_k, \ldots\right) = \; P\left(q_{t+1} = s_j \middle| \, q_t = s_i\right)$$

$$a_{ij} \equiv P\left(q_{t+1} = s_j \middle| \, q_t = s_i\right) = \frac{\#\{transitions \; from \; s_i \; to \; s_j\}}{\#\{transitions \; from \; s_i\}}, 1 < i, j < N \tag{3.5}$$

satisfying $a_{ij} \geq 0$ (3.5) and

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{3.6}$$

So, going from s_i to s_j has the same probability no matter when it happens, or where it happens in the observation sequence. $A = \left[a_{ij}\right]$ is a $N \times N$matrix whose rows sum to 1 (3.6).

From each state s_i, the system moves to s_j with probability a_{ij}, and this probability is the same for any t. Transition probabilities are collected in matrix A.

The Markov model produces a state sequence

$$Q = \{q_1, \dots, q_T\}, q_t \in S \text{ over time } 1 \leq t \leq T$$

A first-order discrete time Markov model, because the probability at $t + 1$ depends only on the states at t.

- *The observation sequence probability*

In an observable Markov model, the states are observable. We have an observation sequence O that is the state sequence $O = Q = \{q_1 q_2 \dots q_t\}$, whose probability is given as

$$P(O = Q|A, \Pi) = P(q_1) \prod_{t=2}^{T} P(q_t|q_{t-1}) = \pi_{q_1} a_{q_1 q_2} \dots a_{q_{T-1} q_T} \quad (3.7)$$

π_{q_1} is the probability that the first state is q_1, $a_{q_1 q_2}$ is the probability of going from q_1 to q_2, and so on. We multiply these probabilities to get the probability of the whole sequence.

Given Π and A, it is easy to generate K random sequences each of length T. We can calculate the probability of an observation sequence is denoted $P(O|A, \Pi)$ (3.7).

3.3 Hidden Markov Models

A Markov chain is useful when we need to compute a probability for a sequence of events that we can observe in the world. In many cases, however, the events we are interested in may not be directly observable in the world. For example, the task of named entity recognition, assigning tags like person, organization, location, date/time/money expressions etc. to words. We didn't observe named entity type; we saw words and had to infer the correct tags from the word sequence. We call the named entity type hidden because they are not observed.

A HMM is a probabilistic sequence model and is shown in Figure 3.4 given a sequence of units (words, letters, morphemes, sentences, whatever), they compute a probability distribution over possible sequences of labels and choose the best label sequence.

In HMM, the states are not observable, but when we visit a state, an observation is recorded that is probabilistic function of the state. We assume a discrete observation in each state from the set $\{v_1, v_2, \dots, v_M\}$.

$$b_j(m) \equiv P\left(O_t = v_m \middle| \, q_t = s_j\right), \; 1 \le j \le N, 1 \le m \le M \tag{3.8}$$

$b_j(m)$ is the observation or emission probability that we observe $v_m, m = 1, \dots, M$ in state s_j (3.8).

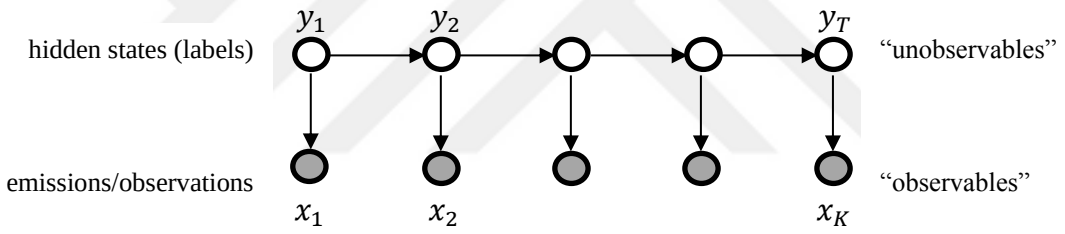

Figure 3.4 Example of a Hidden Markov model

An HMM has the following elements and an example of a HMM which has three states and tree observations is shown in Figure 3.5.

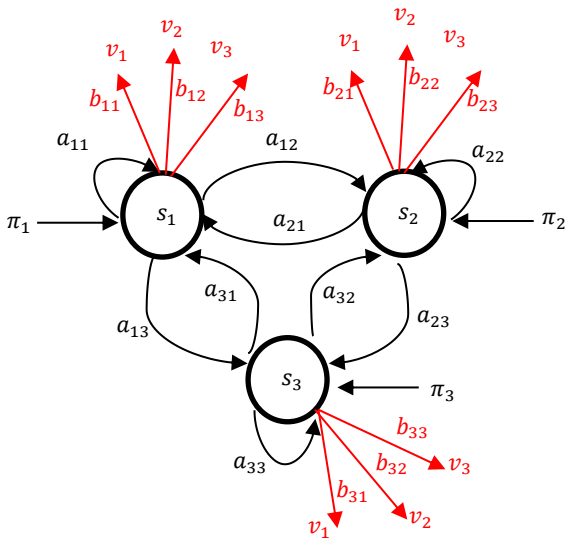

Figure 3.5 Example of a HMM with three states and three observations

- N number of distinct states in the model

$$S = \{s_1, s_2, \dots, s_N\}$$
$$S = \{PER, ORG, LOC, TEM, OTH\}$$

- M number of distinct observation symbols in the alphabet. M is the same a size of the dictionary/vocabulary.

$$V = \{v_1, v_2, \dots, v_M\}$$

- *State transition probabilities*

$$A = [a_{ij}] \text{ where}$$
$$a_{ij} \equiv P(q_{t+1} = s_j | \, q_t = s_i), \qquad 1 < i, j < N$$

- *Observation/Emission probabilities*

$B = [b_j(m)]$ where

$$b_j(m) \equiv P\left(O_t = v_m \middle| \, q_t = s_j\right) = \frac{\#\{observation\ v_m\ in\ a\ s_j\}}{\#\{occurance\ of\ that\ s_j\}}, \quad (3.9)$$

$$1 \leq j \leq N, 1 \leq m \leq M$$

- *Initial state probabilities*

$\Pi = [\pi_i]$ where

$$\pi_i \equiv P(q_1 = s_i)$$

N and M are implicitly define in the other parameters so $\lambda = (A, B, \Pi)$ is taken as the parameter set of a HMM. When given λ, the model can be used to generate a series of arbitrary sequence observations with arbitrary length, but, as always, in the other direction, we are interested in estimating the parameters of the given model in a series of training sequences.

3.4 Graphical Modeling

In some applications such as classifying regions of an image (Li, 2001), estimating the score in a game of Go (Stern, Graepel & MacKay, 2004), segmenting genes in a strand of DNA (Bernal, Crammer, Hatzigeorgiou & Pereira, 2007), syntactic parsing of natural-language text (Taskar, Klein, Collins, Koller & Manning, 2004), we aim to predict an output vector $y = \{y_0, y_1, \dots, y_T\}$ given an observed feature vector x. For POS tagging, y_s is the POS tag of the word at position s, and the input x has feature vectors $\{x_0, x_1, \dots, x_T\}$. Each x_s contains various information about the word in position s, such as its identity, orthographic features such as prefixes and suffixes, membership in domain-specific lexicons etc. Our goal is to maximize the number of labels y_s that are correctly classified. To do this, our system can learn an independent per-position classifier that maps $x \longmapsto y_s$ for each s.

A natural way to represent the way in which output variables are interconnected is provided by graphical models. We use "probabilistic graphical models" to augment the analysis using diagrammatic representations of probability distributions (Bishop, 2006). In other words, graphical modeling is a powerful framework for representation

and inference in multivariate probability distributions (Sutton & McCallum, 2010). A graph contains *nodes* (also called *vertices*) that are linked by *links* (also known as *edges* or *arcs*). In a probabilistic graphical model, each node represents a random variable (or group of random variables) and the links represent the probabilistic relationships between these variables. The graph then captures the distribution of the joint distribution over all random variables, each of which can be decomposed into a product of factors, depending on a subset of variables.

Graphical models separate two major class: *directed* and *undirected*. In directed graphical models (also called as Bayesian networks), the links of the graphs have a particular directionality indicated by arrows. For undirected graphical models (also called Markov random fields), the links do not carry arrows and have no directional significance. Directed models are useful for expressing causal relationships between random variables, whereas undirected models are more suitable for expressing soft constraints between random variables. In order to solve inference problems, it is generally appropriate to convert both directed and undirected models to a different representation called a factor graph.

A factor graph is a bipartite graph $G = (V, F, E)$ in which one set of nodes $V = \{1,2, \dots, |Y|\}$ indexes the random variables in the model, and the other set of nodes $F = \{1,2, \dots, A\}$ indexes the factors.

- "*Bipartite graph*" is a graph shown in Figure 3.6 whose vertices can be divided into two disjoint and independent sets U and V such that every edge connects a vertex in U to one in V (Weisstein, n.d.).

- A "*factor graph*" is a bipartite graph shown in Figure 3.6 representing the factorization of a function (Kschischang, Frey & Loeliger, 2006). Factor graphs are used to represent factorization of a probability distribution function enabling efficient computations.

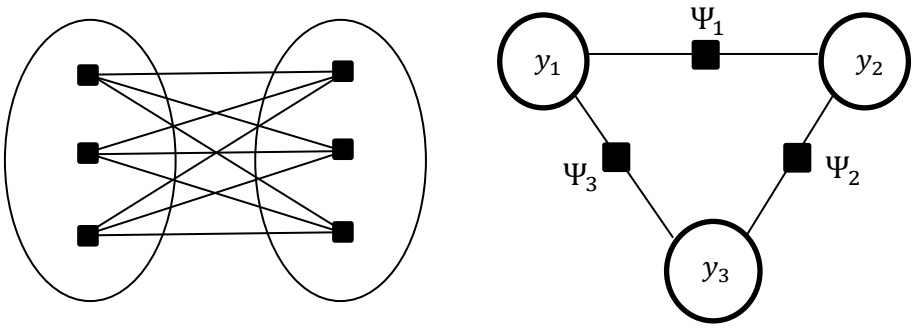

Figure 3.6 An example of a bipartite graph (left-side) and factor graph with three variables (right-side)

The absence of an edge between two variables represents conditional independence between variables into model. *Conditional independence* means that two random variables a and b are independent given a third random variable c. If they are independent, conditional probability distribution is calculated by 3.10. Conditional independence does not induce the absence of the edge. This type of graph is called independency graph. It does not contain any information about the probability distribution, only the absence of the edge is informative. For each conditional distribution, we add directed links (arrows) to the graph from the nodes corresponding to the variables conditioned to the distribution.

$$p(a, b|c) = p(a|b)p(b|c) \tag{3.10}$$

Directed and undirected graphical models differ in the way that the original probability distribution is factorized. The factorization into a product of conditional probability distributions is done as in a directed graphical model. In undirected graphical models, the factorization is done into arbitrary functions. This does not require an explicit specification how the variables are related. In this model, calculating the normalization factor is expensive.

3.4.1 Directed Models

Whereas the local functions in an undirected model need not have a direct probabilistic interpretation, a *directed graphical model* describes how a distribution factorizes into local conditional probability distributions. Directed models are often

used as generative models. Let G be a directed acyclic graph, in which $\pi(s)$ are the indices of the parents of each node y_s in G. A directed acyclic graph is a finite directed graph with no directed cycles. A directed graphical model is a family of distributions that factorize as in (3.11).

$$p(y) = \prod_{s=1}^{S} p\left(y_s \middle| y_{\pi(s)}\right) \tag{3.11}$$

We refer to the distributions $p\left(y_s \middle| y_{\pi(s)}\right)$ as local conditional distributions. Note that $\pi(s)$ can be empty for variables that have no parents. In this case $p\left(y_s \middle| y_{\pi(s)}\right)$ should be understood as $p(y_s)$. Figure 3.7 represents an HMM classifier for a sequence of three input variables (Klinger & Tomanek, 2007).

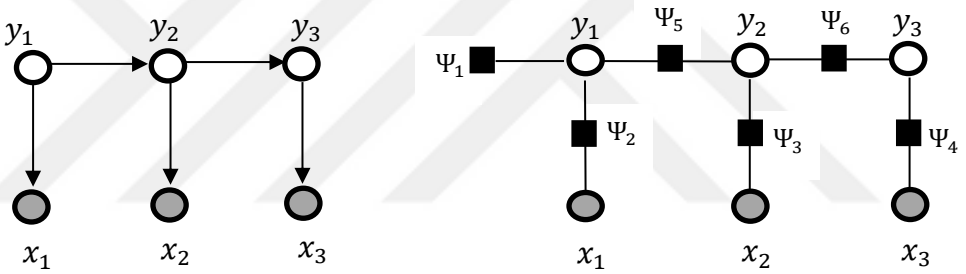

Figure 3.7 Independency (left-side) and factor (right-side) graph for the HMM

3.4.2 Undirected Models

A probability distribution p of interest can be represented by a product of factors of the form $\Psi_a(y_a)$ where a is an integer index that ranges from 1 to A, the number of factors. Each factor Ψ_a depends only on a subset $y_a \subseteq Y$ of the variables. The value $\Psi_a(y_a)$ is a non-negative scalar that can be thought of as a measure of how compatible the values y_a are with each other. This factorization can allow us to represent p much more efficiently, because the sets y_a may be much smaller than the full variable set Y.

An *undirected graphical model* is a family of probability distributions that each factorize according to a given collection of factors. Formally, given a collection of

subsets $\{Y_a\}_{a=1}^{A}$ of Y, an undirected graphical model is the set of all distributions that can be written as (A, the number of factors)

$$p(y) = \frac{1}{\mathrm{Z}} \prod_{a=1}^{A} \Psi_a(y_a), \tag{3.12}$$

for any choice of factors $\mathcal{F} = \{\Psi_a\}$ that have $\Psi_a(y_a) \geq 0$ for all y_a. (The factors are also called *local functions* or *compatibility functions*.) We will use the term *random field* to refer to a particular distribution among those defined by an undirected model.

The constant Z is a normalization factor that ensures the distribution p sums to 1. It is defined as in (3.13):

$$\mathrm{Z} = \sum_{y} \prod_{a=1}^{A} \Psi_a(y_a) \tag{3.13}$$

The quantity Z, considered as a function of the set $\mathcal{F}$ of factors, is also called the *partition function*.

3.5 Conditional Random Fields

CRF is derived from Markov network and is introduced by (Lafferty, McCallum & Pereira, 2001). CRF is a special case of undirected graphical models, also known as Markov Random Fields (MRFs). CRF is a sequence data labeling recognition model based on statistical approaches. CRF uses an undirected graphical model to calculate the conditional probability $p(\vec{y}, \vec{x})$. CRF are probabilistic models for computing the probability $p(\vec{y}, \vec{x})$ of a possible output $\vec{y} = (y_1, y_2, \ldots, y_{\mathrm{n}})$ for given the input $\vec{x} = (x_1, x_2, \ldots, x_{\mathrm{n}})$ which is also called the observation. The general model formulation of CRF's is derived as:

$$p(\vec{y},\vec{x}) = \frac{1}{Z(\vec{x})}\prod_{a\in A}\psi_a(\vec{x}_a,\vec{y}_a) \tag{3.14}$$

with

$$Z(\vec{x}) = \sum_{\vec{y}}\prod_{j=1}^{n}\psi_j(\vec{\mathrm{x}},\vec{y}) \tag{3.15}$$

CRFs have different types: general, linear-chain, semi-Markov, graphical, grid-shaped, tree-shaped, skip-chain, etc. A special form of a CRF, which is structured as a linear chain, models the output variables as a sequence is shown in Figure 3.8.

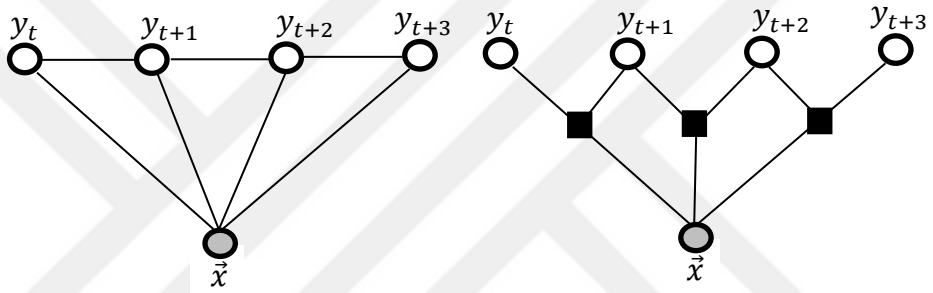

Figure 3.8 Independency (left) and factor (right) graph for the linear chain CRF

Given the factors $\psi_j(\vec{\mathrm{x}},\vec{y})$ in the form

$$\psi_j(\vec{\mathrm{x}},\vec{y}) = \exp\left(\sum_{i=1}^{m}\lambda_i f_i\left(y_{j-1},y_j,\vec{x},j\right)\right) \tag{3.16}$$

and assuming $n+1$ to be length of the observation sequence, a linear chain CRF formula can be written as

$$p_{\vec{\lambda}}(\vec{y}|\vec{x}) = \frac{1}{Z_{\vec{\lambda}}(\vec{x})}\exp\left(\sum_{j=1}^{n}\sum_{i=1}^{m}\lambda_i f_i\left(y_{j-1},y_j,\vec{x},j\right)\right) \tag{3.17}$$

In (3.17), j specifies the position in the input sentence $\vec{x}$. The weights λ_i are not dependent on the position j. This technique, known as parameter tying, is applied to ensure a specified set of variables to have the same value.

The normalization to [0,1] is given by

$$Z_{\vec{\lambda}}(\vec{x}) = \sum_{\vec{y} \in Y} \exp\left(\sum_{j=1}^{n} \sum_{i=1}^{m} \lambda_i f_i(y_{j-1}, y_j, \vec{x}, j) \right) \tag{3.18}$$

3.5.1 Training

For all types CRFs, as well as for MEMs, the maximum-likelihood method can be applied for parameter estimation. That means, that training the model is done by maximizing the log-likelihood on the training data. So, training means finding them λ parameters. For this we need fully labeled data sequences $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$ where $x^{(1)} = x_{1:N_1}^{(1)}$ the first observation sequence, and so on. Since CRFs define the conditional probability $p(y|x)$, the appropriate objective for parameter learning is to maximize the conditional likelihood of the training data

$$\sum_{j=1}^{m} \log p(y^j | x^j) \tag{3.19}$$

Often one can also put a Gaussian prior on the λ's to regularize the training (i.e., smoothing). If $\lambda \sim N(0, \sigma^2)$, the objective becomes

$$\sum_{j=1}^{m} \log p(y^j | x^j) - \sum_{i}^{F} \frac{\lambda_i^2}{2\sigma^2} \tag{3.20}$$

The standard parameter learning approach is to compute the gradient of the objective function, and use the gradient in an optimization algorithm like L-BFGS.

3.5.2 Inference

The inference problem is to find the most likely sequence $\vec{y}$ for given observations $\vec{x}$. This is not to select a set of states that are most likely to be individually. This will be the maximization of the number of correct situations in the sequence. The Viterbi algorithm is applied to find the most likely sequence (Rabiner, 1989). The Viterbi algorithm is similar to the Forward-Backward algorithm. The main difference is to apply maximization instead of summing.

The quantity $\delta_j(s|\vec{x})$, which is the highest score along a path, at position j, which ends in state s, is defined as

$$\delta_j(s|\vec{x}) = \max_{y_1,y_2,\dots,y_{j-1}} p(y_1, y_2, \dots, y_j = s|\vec{x}) \tag{3.21}$$

The induction step is

$$\delta_{j+1}(s|\vec{x}) = \max_{s' \in S} \delta_j(s').\psi_{j+1}(\vec{x}, s, s') \tag{3.22}$$

CHAPTER FOUR
THE PREPROCESS, ANNOTATION AND FEATURE EXTRACTION STAGES

4.1 Datasets

Annotated data is an important requirement when you study with machine learning based approaches in natural language domain. The annotation stage is realized manually by humans who have knowledge about the domain or data. But human annotation is costly process. A major obstacle to NER on Turkish is the scarcity of publicly available annotated corpus and morphological processing tool.

In this thesis, we introduced a new annotated dataset for Turkish NER and named it as METU-NER, which includes our annotation of a part of METU Turkish Corpus (Say et.al, 2002). Beside, we also used another dataset (we called ITU-NER) (Seker & Eryigit, 2017), which includes four versions: one is used for training set (TRAIN7) and rest are testing purposes (WFS7, IWT, TDS). We performed experimentation with both METU-NER and ITUNER, where some properties such as tokens count and entity distribution are given in Table 4.1.

TRAIN7 and WFS7 includes newspaper articles with nearly 500K tokens, and annotated for ENAMEX types (Tür et al., 2003). This dataset contains well written text. It is also the largest dataset and widely used in Turkish NER researches. It is re-annotated to extend the covered named entity types to TIMEX and NUMEX in (Seker & Eryigit, 2017). Tweet Dataset (TDS) is includes 55K twit messages Twitter corpus and type of user generated content (Çelikkaya et al., 2013). TDS is collected from Twitter and consists many exceptions and spelling error. It is re-annotated to improve the consistency and the quality of the annotations by MUC-6 guidelines.

Sulubacak, Torunoğlu-Selamet & Eryiğit (2015) annotated a Turkish treebank from the social media domain: ITU Web Treebank (IWT). IWT includes sentences in Turkish collected from internet which are comments, reviews and posts. However,

web users generally write spoken words in their generated content and they don't pay attention to correct grammar, it is full of spelling mistakes, omitting letters or replacing them with foreign characters etc.

Table 4.1 The details of statistics for datasets

	ITU-NER				**METU-NER**
NE type / Desc.	**TRAIN7**	**WFS7**	**TDS**	**IWT**	
Total # of Token	444,843	24,753	46,406	43,225	628,698
Person	14,693	658	4,259 (3562 contains @)	380	15,011
Organization	9,160	410	424	401	8,600
Location	9,767	637	184	260	10,845
Time	148	19	20	9	3,018
Date	1,366	119	56	59	
Money	597	41	-	45	-
Percentage	648	62	1	8	-
TOTAL NE	**36,379**	**1,946**	**4,944**	**1,162**	**37,474**

METU Turkish Corpus is a collection of approximately 2 million words. It has 998 XCES-tagged files on different types such as novel, story, research-survey, article, travel, interview, memoir, news, column and essay. The distribution of the documents in the news genre is 42% in the corpus. The each XCES-tagged file has a format of xml based, include informative tags such as the author of the text, the paragraph boundaries in the text, etc. The XCES is corpus encoding standard for xml. Firstly, we stripped these informative tags to obtain the pure content. Then we manually tagged 324 documents from METU-NER with the support of 10 annotators, they are all undergraduate students interns working on our research lab. The documents were selected according with their weight in the count of genre distribution rate: news 45.37% (147 documents); columns 11.42% (37 documents); story 7.1% (23 documents); novel and essays 6.48% (21 documents) evenly; article 5.25% (17 documents); research-survey 4.63% (15 documents), memoir and travel 1.54% (5 documents) evenly; interview 1.23% (4 documents) and other 8.95% (29 documents). Tagging was performed using Brat Rapid annotation tool which is an intuitive web

based tool for text annotation supported by NLP technology (Stenetorp, Pyysalo & Topi, 2012). While annotating, we considered that the apostrophes and the suffixes are part of NEs. In Some dataset consider the apostrophes and the sequence of suffixes attached to NEs as part of NE types or not. This issue is outstanding and some systems consider part of NEs while others exclude them during annotation (Doğan Küçük et al., 2017).

There are many more types of NEs than the three classical types in ENAMEX. Sekine, Sudo & Nobata, (2002) proposed the extended named entity hierarchy, which has 242 NE types. We handled this hierarchy as guideline for deciding these categories. We presented this hierarchy as Turkish in our other study (Öztürkmenoğlu, 2012). Although we aimed to extract temporal type NEs at the beginning of the study, we observed that there was confusion in the annotation step due to the difference semantic interpretation or be less careful of annotators. Therefore, by removing the class distinction in the temporal NEs, we reduced the NE type diversity to four types: person, organization, location and temporal. Additionally, we labeled other tokens as "other" which may belong to any other NEs which we don't consider.

METU-NER contains 55,359 sentences and 628,698 tokens, in total. 37,474 NEs (53,352 tokens) were tagged in four categories: 15,011 person names (PER), 8,600 organization names (ORG), 10,845 location names (LOC) and 3,018 temporal expressions (TEM). In annotation results, 25,567 annotated words contain only a token and 11,907 annotated words contain multi tokens. The annotation statistics are given in the Table 4.2.

We prepared the guide file to annotators for helping annotation. We hosted brat annotation tool in our web server. The annotation tool saved the annotation's results with boundaries of annotated named entity. The exactly matching of annotation's position with token's position is important so we should be careful in tokenization step. Figure 4.3. shows a sample annotation result, which was taken as output from annotation. The columns are shown respectively: document name, NE type, start position of annotation, end position of annotation and annotated words.

Table 4.2 The distribution of named entity types in METU-NER

NE type (short name) (hierarchy number)	Total # of annotated NEs (Total # of tokens)	Annotation example
Person names (PER)	15,011 (21,597)	"Sait Faik'i", "Peyami Safa'nın", "Rauf Denktaş'ın", "Francis D.K. Ching'in", "Fatih Terim,", "Atatürk'ümüz...", "Ahmet Necdet Sezer'in"
Organization names (ORG)	8,600 (14,471)	"Sağlık Bakanlığı'nı,", "Bakanlar Kurulu'nu", "Hacettepe Tıp Fakültesi", "İMKB", "Galatasaray", "Türkiye Büyük Millet Meclisi'ni", "Dokuz Eylül Üniversitesi'nden", "Dr. Faruk Sükan Doğumevi Hastanesi'ndeki","DYP Genel Başkanlığı'na"
Location names (LOC)	10,845 (12,176)	"İstanbul Boğazı'ndaki", "'Dumlupınar'ın", "İrlanda'nın", "Yunanistan'da", "Yenikapı Sahil Kennedy Caddesi'nde", "Yahya Kaptan Semti'nde", "Virjin Adaları,", "Şeyhnasır Dağı'nın"
Temporal expressions (TEM)	3,018 (5,108)	"XIX. yüzyıl", "16-20 Temmuz 1994'de", "1918-20'lerde", "saat 20.30.", "Mart 1994'te", "14.10.2002", "28 Şubat", "20:12'den", "2005"
Total	**37,474 (53,352)**	

Table 4.3 Sample annotation output for document "23710000"

NE type	Start pos.	End pos.	Annotated words
ORGanization	67	86	Çorum İş Mahkemesi,
ORGanization	197	217	Anayasa Mahkemesi'ne
ORGanization	260	272	Çorum Barosu
PERson	288	301	Teoman Şahin,
TEMporal	1943	1953	03.01.2003
LOCation	2164	2171	Türkiye
PERson	2620	2636	Gündaş Alaşahan,
LOCation	3180	3193	Şanlıurfa'nın
ORGanization	6511	6561	Karayolları ve Köy Hizmetleri Bölge Müdürlüklerine
TEMporal	7393	7406	saat 10:35'te
TEMporal	11609	11620	8 Ocak 2003

4.2 Tokenization

In METU-NER dataset, we used dot ('.'), question mark ('?') and exclamation mark ('!') as delimiters for sentence segmentation. After sentence segmentation, we tokenized data using a white-space characters and we didn't separate any punctuation characters from the left or right. We supposed the usage together the Carriage Returns and Line Feed characters ('\r\n') as delimiters for paragraph segmentation. In ITU-

NER datasets, all punctuation marks are considered as a token and they have no paragraph information and their sentences are separated by an empty line. After tokenization, we stored all token information in PostgreSQL (Stonebraker, Rowe & Hirohama, 1990) database, together with their document number, paragraph and sentence indexes, start and end character positions, NE type and chunks encoding information.

Chunk is a part of named entity. In the studies, chunks are encoded in some formats as taggers, base formats are IO, BIO and BMEWO ("Coding Chunkers as Taggers: IO, BIO, BMEWO, and BMEWO+ | LingPipe Blog", 2009). IO encoding is the simplest encoding which tags each token as either being in (I_X) a particular type of named entity type X or in no entity (O). In BIO encoding, B shows begin-of-entity (B_X) and I shows continuation-of-entity (I_X). In the BMEWO encoding, end-of-entity tokens are shown with E_X and mid-entity tokens are shown with M_X, and a whole new tag for a single-token entities are shown W_X. The system encoding format is an open issue for NER researches (Doğan Küçük et al., 2017). In our experiments, we compared the effects of encoding format. Table 4.4 shows an example of a tokenized string, its stored information in database and its encoding using several standards. For example; *"Akgün Petrol İstasyonu'nun" (Akgün Petrol Station)* is an organization NE. It has 3 tokens *"Akgün"*, *"Petrol"* and *"İstasyonu'nun"*. In RAW and IO encoding standards, all of tokens are shown with same value as ORG in RAW, I_ORG in IO encoding. In BIO encoding, the first token *"Akgün"* is shown with B_ORG and the other tokens are shown with I_ORG. The first token *"Akgün"* is shown with B_ORG, the last token *"İstasyonu'nun"* is shown with E_ORG and the other tokens "Petrol" between first and last are shown with M_ORG.

Table 4.4 An example of tokenized sentence and its output encoding standards

Token	Start	End	Index	RAW	IO	BIO	BMEWO
Menemen	1070	1077	125	LOC	I_LOC	B_LOC	B_LOC
İlçesi'ndeki	1078	1090	126	OTH	O	O	O
Akgün	1091	1096	127	ORG	I_ORG	B_ORG	B_ORG
Petrol	1097	1103	128	ORG	I_ORG	I_ORG	M_ORG
İstasyonu'nun	1104	1117	129	ORG	I_ORG	I_ORG	E_ORG
sahibi	1118	1124	130	OTH	O	O	O

Table 4.4 continues

Muhlis	1125	1131	131	PER	I_PER	B_PER	B_PER
Akgün	1132	1137	132	PER	I_PER	I_PER	E_PER
ile	1138	1141	133	OTH	O	O	O
pompacı	1142	1149	134	OTH	O	O	O
Gürsel	1150	1156	135	PER	I_PER	B_PER	B_PER
Tekin,	1157	1163	136	PER	I_PER	I_PER	E_PER
teşhis	1164	1170	137	OTH	O	O	O
için	1171	1175	138	OTH	O	O	O
Ödemiş'e	1176	1184	139	LOC	I_LOC	B_PER	W_LOC
getirildi.	1185	1195	140	OTH	O	O	O

4.3 Morphological Analysis

In morphologically rich languages, like Turkish, many surface forms can be generated from the same stem as a different training element or feature. This state causes data sparseness problem. To overcome this problem, morphological level processing is a requirement for Turkish NER. The outputs of morphological analysis (stem of word, inflectional and derivational tags) are used as features in CRF model. The outputs of morphological analysis may produce more than one analysis results, in this state, the morphological disambiguator executes on these outputs to select the most probable usage in the given context. In Turkish, some research groups focusing natural language processing research are working on morphological analysis (Akın & Akın, 2007; Eryiğit, 2014; Oflazer, 1993; Sahin, Sulubacak & Eryigit, 2013) and disambiguation (Eryiğit, 2014; Sak et al., 2008) particularly. It is not possible to access many of these types of Turkish NLP tools because the researches continue or it is not allowing unlimited usage to for a large number of tokens.

In this thesis, we used a two-level morphological analyzer (Oflazer, 1994). However, this tool does not process tokens containing numeric letter, and punctuation characters (except apostrophe sign) are removed from the token at the end position. For morphological disambiguator, we selected the analysis result with the most number of POS tags from the results of the analysis of each word. If it is equal, we have chosen the result with the longest analysis. Table 4.5 gives an example output of morphological analyzer and disambiguator. For example, a word "*taşınmazların* (*your*

immovables)", the morphological analyzer returned six results. The number of result for each distinct POS tag is equal. In this state, we selected the longest length of morphemes after splitting '+' (plus) sign. So the bold row in Table 4.5 was taken the result of morphological analysis *(taşı+Verb^DB+Verb+Pass+Neg^DB+Adj+AorPart^DB+Noun+Zero+A3pl+P2sg +Nomin)*[1] for sample.

Table 4.5 An example result of morphological analyzer and disambiguator for "*taşınmazların*"

Stem	**POS tag**	**Morphemes**	**# of morph parts**
taşı	*Verb^DB*	*Verb+Pass+Neg^DB+Adj+AorPart^DB+Noun+Zero+A3pl+Pnon+Gen*	10
taşınmaz	Noun	A3pl+Pnon+Gen	3
taşın	Verb	Neg^DB+Adj+AorPart^DB+Noun+Zero+A3pl+P2sg+Nom	8
taşı	***Verb^DB***	***Verb+Pass+Neg^DB+Adj+AorPart^DB+Noun+Zero+A3pl+P2sg+Nom***	10
taşınmaz	Noun	A3pl+P2sg+Nom	3
taşın	Verb	Neg^DB+Adj+AorPart^DB+Noun+Zero+A3pl+Pnon+Gen	8

4.4 The Extended Feature Set and Feature Extraction

Features are descriptors or characteristic attributes of words designed for algorithmic consumption. The features can be generated from hand-crafted rules (obeyed the rules or not), database lookup (existence in a dictionary or not) operations, linguistic or syntactic rules (the right usage of capital letter or apostrophe) or extra features for NE type to be extracted such as date, time, money, percentage expressions.

These features may take binary, numeric or nominal values. If a word is capitalized, the feature value is *true* or *false* otherwise. A numeric attribute corresponding to the length of token and a nominal attribute to the case version of the token such as

[1] The abbreviations after the plus sign stand for +Verb^DB: Verb derivation boundary, +Verb: Verb, +Pass: Passive,+Neg^DB: Negative derivation boundary, + Adj: Adjective, +AorPart^DB: Express perfective refer past, +Noun: Noun, +Zero: Zero derivation with no overt derivation, +A3pl: 3rd person plural aggrement, +P2sg: 2nd person singular aggrement, +Nom: Nominative case

lowercase, uppercase or mixed case. In order to use on the CRF model, we extracted the following best-fitting features into four categories: rule based, lexical, dictionary lookup and morphological.

4.4.1 Rule Based Features

Rule based features (RBF) are binary features. For example, the main rules governing named entities of type person, location, and organization names in Turkish are provided. One of them indicates if the initial letters of all tokens of the named entities are capitalized, or not. The second one, the sequence of inflectional suffixes added at the ends of the named entities is separated from them with an apostrophe. Considering such rules, we defined some rules using regular expressions or regex, are indicated by r and are given in Table 4.6. We applied them on surface form of token which is indicated by t. Some regex rules such as *allLetterUppercase, allLetterLowercase, letterMixedcase* were applied on tokens which were stripped from numeric and punctuation marks with *regexp_replace* function. We applied regex rules on inputs using *regexp_matches* function. The generalization formula of this case is shown below in 4.1. When the given states are met, they get true (1) value, otherwise false (0) value. In Table 4.6, the second column for each rule shows the short code using in our system. The rules of 20,21 and 22 are presented as extra features for TIMEX and NUMEX types in Seker and Eryigit (2017)'s study.

$$f(t,r) = \begin{cases} 1 \text{ , if } t \text{ matches with } r \\ 0 \text{ , other case} \end{cases} \quad (4.1)$$

- **RBF1:** If the token contains only uppercase letter and then dot sign.
- **RBF2:** If the token contains only uppercase letter.
- **RBF3:** If the initial letter of token is in upper case.
- **RBF4:** If the all letters of token are in upper case.
- **RBF5:** If the all letters of token are in lower case.
- **RBF6:** If the token contains together upper and lower case letters.
- **RBF7:** If the token contains only digits.

- **RBF8:** If the token contains least a digit.
- **RBF9:** If the token ends with dot sign.
- **RBF10:** If the token ends with comma sign.
- **RBF11:** If the token ends with punctuation marks (without apostrophe character).
- **RBF12:** If the token contains any punctuation marks (without apostrophe character).
- **RBF13:** If the token contains the dash character.
- **RBF14:** If the token contains the slash character.
- **RBF15:** If the token contains the apostrophe character.
- **RBF16:** If the sequence of characters in token is like date expression.
- **RBF17:** If the sequence of characters of token is like time expression.
- **RBF18:** If the token is like a web site address (i.e. http://fbe.deu.edu.tr).
- **RBF19:** If the token is like an e-mail address (i.e. http://fbe.deu.edu.tr).
- **RBF20:** If the token contains the percentage character.
- **RBF21:** If the token starts with the term of '*saat*'.
- **RBF22:** If the token contains the column indicator character.

Table 4.6 The details of regular expressions used in rule based features

#RBF	Description	Regex	Example
1	singleUppercaseWithDot	^[A-ZİĞÜŞÖÇÂÊÎÛÉ][.]$	Ş.
2	singleUppercase	^[A-ZİĞÜŞÖÇÂÊÎÛÉ]{1}$	T
3	firstLetterUppercase	^[A-ZİĞÜŞÖÇÂÊÎÛÉ]+.*$	Osmanlı
4	allLetterUppercase	^[A-ZİĞÜŞÖÇÂÊÎÛÉ]+$	TBMM
5	allLetterLowercase	^[a-zğıüşöçâêîûé]+$	teknoloji
6	letterMixedcase	^(?=.*[a-zğıüşöçâêîûé])(?=.*[A-ZİĞÜŞÖÇÂÊÎÛÉ])[a-zğıüşöçâêîûéA-ZİĞÜŞÖÇÂÊÎÛÉ]+$	Anadolu
7	onlyDigit	^[\d]+$	2002
8	leastAdigit	^.*[\d].*$	1943'e
9	finishWithDot	^.{1,}[.]$	hesaplanıyor.
10	finishWithComma	^.{1,}[,]$	Atatürk,
11	finishWithPunctuation	^.{1,}[\^\.,:;?!"%#*&$@=\[\]{}\(\)\/\<\>“-”—_\\+><»·…]$	Hititlere,
12	containsPunctuation	^.*[\^\.,:;?!"%#*&$@=\[\]{}\(\)\/\<\>“-”—_\\+><»·…].*$	(TÜSİAD)

Table 4.6 continues

13	containsDash	^.*[-].*$	televizyon-radyo
14	containsSlash	^.*[/].*$	25-27/Ekim/2002'de,
15	containsApostrophe	^.*[`"'‘].*$	Ankara'da
16	hasDate	^(0?[1-9]|[12][0-9]|3[01])["-,/](0?[1-9]|1[012])["-,/](d{4}|d{2})+.*$	37373
17	hasClock	^([01]?[0-9]|2[0-3])[.:]([0-5][0-9])+.*$	10,57
18	isURL	^(http:\/\/www\.|https:\/\/www\.|http:\/\/|https:\/\/)?[a-z0-9]+([\-\.]{1}[a-z0-9]+)*\.[a-z]{2,5}(:[0-9]{1,5})?(\/.*)?$	www.sanalkutuphane.net
19	isEmailAddress	^[A-Za-z0-9._%-]+@[A-Za-z0-9.-]+[.][A-Za-z]+$	ismetsalu2003@yahoo.com
20	containsPercentageSign	^.*[%].*$	99.9%
21	containsOclockTerm	^saat.*$	saatlerinde
22	containsColumnIndicatorSign	^.*[:].*$	Bilge'ye:

4.4.2 Lexical Features

Lexical features (LF) are extracted via user defined database functions. Some operators and functions are used in these functions such as *char_length, array_length, regexp_replace, regexp_matches, min* and *max* etc. The some features (LF2, LF5 and LF6) are defined below are presented in Seker and Eryigit (2017)'s study.

LF1 - *get_char_length*: We calculate a length of the token using *char_length* function. It is indicated using "*charlen*" prefix to length value in a sequence of token. For token "*Atatürk*,", this feature is "*charlen8*".

LF2 - *get_numeric_pattern_value*: This feature is used for numeric tokens and is extracted using regex that is applied on tokens. We used 1 for integer tokens between [1,12], 2 for integer tokens between [13,31], 3 for integer tokens between [32,2020], 4 for other integer tokens and 5 for non-numeric tokens. After extracting the value, this feature is indicated by "*nv*" prefix to value such as "*nv4*" for token of "1923".

LF3 - *position_in_sentence*: We used the "*B*", "*I*", "*E*" tags with "*position*" prefix as the value of this property, depending on whether the current token was at the beginning, end, or middle of a sentence. For example, for this feature, a token of

"*Türkiye'nin*" is shown as "*positionB*", "*Ankara'daki*" is shown as "*positionI*" and "*sürüyor*" is shown as "*positionE*" which these tokens are existed in this sample sentence: "*Türkiye'nin Bağdat Büyükelçisi ve Irak'ın Ankara'daki Büyükelçisi aracılığıyla görüşmelerimiz sürüyor.*".

LF4 - *get_pattern_of_surface*: When extracting the value of this feature, we aim to create a pattern template based on the letters in each token. We changed uppercase letters with "*X*" letter, lowercase letters with "*x*" letter, numeric letters with "*d*" and punctuation marks with "*p*" letter into each token. For example, the tokens "*2002'de*", "*Ankara'da*" is shown as "ddddpxx", "Xxxxxpxx" respectively.

LF5 - *g_case_feature*: If all the letters of a token consist of uppercase letters, we used the tag "*UPPERCASE*" and "*LOWERCASE*" if all the letters were lowercase. If the token contains at least both uppercase and lowercase letters, we used the tag "*MIXEDCASE*". If the initial letter in the token is capital, we used the "*PROPERCASE*" tag.

LF6 - *g_position_in_sentence*: This is similar to the LF3 feature. Unlike LF3 feature, we are interested in whether the current token is the first word of the sentence. Considering the example sentence in LF3, "*Türkiye'nin*" is shown as "*positionB*" and the other tokens are shown as "*positionNotB*".

4.4.3 Dictionary Lookup Features

The two types of dictionaries, (base and generated) for different NE genres that were created by collecting from different sources, are used. The records in the dictionaries are separated by lines and may include multiple token records. Dictionary lookup features (DLF) are binary features. If the token (stem of the word or surface form of word) exists in the corresponding dictionary, feature has a value of 1 or 0 otherwise. When searching in dictionaries consists multiple token records, we created a recursive method in which the consecutive token is searched sequentially and the search process continues as the result returns. In this method, we removed apostrophe and its right

context from tokens so we used nearly surface forms of word. In this way, when we analyzed the records in basic first person names (“*AHMET HAMDİ”*, “*YAVUZ SELİM”* etc.), surnames (“*ÖĞER YAMAN*”, “*ÇETİN ALTAY*” etc.) and location (“*UĞUR MUMCU MAH.*”, “*PAPUA YENI GINE*” etc.) dictionaries, we found that the results would be more successful. When searching in dictionaries with multiple tokens, a match between one of the tokens in the dictionary and the stem of the current token was searched. The generator dictionaries, which are rather smaller than base dictionaries. The records in generator dictionaries consists of some frequently pre-words or suffix words to derive NEs. Table 4.7 provides the number of tokens in each dictionary. In dictionary lookup process, we pay attention to the encoding standard (UTF-8) and converting the tokens to uppercase letters considered the conversion of Turkish letters and so normalize them.

Table 4.7 The count of distinct tokens in dictionaries

#DLF	Dictionary name	Type	Has multi token	# of tokens
1	First name, Surname *(İsim, Soyisim)* [2] [3] [4] [5] [6]	base	no	6,722
2	Gazetteer (*Coğrafi Yer*)[7][8][9][10]	base	yes	35,218
3	Abbreviations *(Kısaltmalar)*[11]	base	yes	1,043
4	Occupations *(Meslekler)*[12]	base	yes	734
5	Temporal *(Tarih,Zaman)*	generated	no	42
6	Numeric *(Sayı,Sıralama)*	generated	no	61
7	Currency *(Para Birimleri)*	generated	no	11
8	Pre- and Suf- Words *(Ön ve Son Ek Kelimeler)*	generated	no	172
9	First name *(İsim)*	base	yes	86,561
10	Surname *(Soyisim)*	base	yes	146,923
11	Gazetteer *(Coğrafi Yer)*	base	yes	38,079
12	Month Names *(Ay İsimleri)*	base	no	12
13	Title *(Ünvan)*	generated	no	22

[2] https://gist.github.com/emrekgn/b4049851c88e328c065a
[3] https://tr.wiktionary.org/w/index.php?title=Kategori:Türkçe_erkek_adları
[4] https://tr.wiktionary.org/wiki/Kategori:Türkçe_kız_adları
[5] https://gist.github.com/emrekgn/493304c6445de15657b2
[6] https://tr.wiktionary.org/w/index.php?title=Kategori:Türkçe_soyadlar
[7] http://postakodu.ptt.gov.tr/Dosyalar/pk_list.zip
[8] http://www.tanitma.gov.tr/TR,22861/ulke-telefon-kodlari.html
[9] https://tr.wikipedia.org/wiki/Ülkeler_listesi
[10] https://help.surveymonkey.com/articles/tr/kb/Do-you-have-a-list-of-countries
[11] http://www.tdk.gov.tr/index.php?option=com_content&id=198:Kisaltmalar
[12] https://tr.wikipedia.org/wiki/Meslekler_listesi

Table 4.7 continues

14	Location Generator *(Yer Türeteç)*	generated	no	45
15	Organization Generator *(Kurum Türeteç)*	generated	no	60
16	Currency *(Para Birimleri)*	generated	no	29

In Table 4.7, the dictionaries between DLF9-DLF16 were taken from Seker and Eryigit (2017)'s study. We have gathered the records are stored into between DLF1-DLF8 from different web resources and imported them distinctly into dictionaries. Some generator dictionaries are generated manually which are used mostly frequent words with NEs as prefix or suffix words such as "*TL*", "*dolar*", "*kuruş*", "*Eylül*", "*Perşembe*", "*yetmişinci*", "*onaltı*", "*milyon*", "*dernek*", "*fakülte*", "*mahalle*", "*sokak*", "*kulüp*", etc. In Table 4.8 shows the expanded dictionaries by combining dictionaries starting with DLF1-DLF8 and DLF9-DLF16 in uniquely.

Table 4.8 The count of distinct tokens in dictionaries

#DLF	**Dictionary name**	**Type**	**Has multi token**	**Count of tokens**
17	First name *(İsim)*	base	yes	98,628
18	Surname *(Soyisim)*	base	yes	151,445
19	Abbreviations *(Kısaltmalar)*	base	yes	1,045
20	Gazetteer (*Coğrafi Yer*)	base	yes	65,391
21	Occupations *(Meslekler)*	base	no	586
22	Location Generator *(Yer Türeteç)*	generator	no	100
23	Organization Generator *(Kurum Türeteç)*	generator	no	146
24	Temporal *(Tarih,Zaman)*	generator	no	49
25	Numeric *(Sayı,Sıralama)*	generator	no	61
26	Currency *(Para Birimleri)*	generator	no	36
27	Organizations *(Kurumlar)*	base	yes	3,186

4.4.4 Morphological Features

Morphological features (MF) are extracted from analysis produced after the morphological analysis and disambiguation processing of each token. We emphasized the importance of this process and open research issue according to morphological rich languages in previous section.

MF1 – *ko_stem*: The stem information is extracted from morphological analysis of token is introduced in Section 4.3.

MF2 – *ko_pos*: In morphological sequence, the first part of analysis after the stem is considered to the value of this feature as Part-of-Speech (POS) tag.

MF3 – *ko_morpheme*: All inflectional tags after the POS tag. For example, the feature value will be "*taşı+Verb+Pos^DB+Adj+PresPart*" for the word "*taşıyan* (*bearing*)" with the morphological analysis.

MF4 – *ko_prop*: The "+Prop" tag exists in morphological analysis or not. It is indicated as binary feature 1 or 0.

MF5 – *ko_nouncase*: If a morphological analysis contains one of the following values: Nominative (NOM), Accusative (ACC), Dative (DAT), Ablative (ABL), Locative (LOC), Genitive (GEN), Instrumental (INS) and Equative (EQU), this feature is 1 and otherwise 0.

MF6 – *normalized_form*: We converted all letters in surface form of token to lower case for normalizing tokens.

MF7 – *g_pos*: The last part of analysis in morphological analysis after the stem is considered to the value of this feature. In morphologically rich languages, the usage of derivations may change their part of speech categories. The final form determines its POS-tag in a sentence. If the tokens separated by apostrophe from the proper nouns, this feature value is assigned with a special POS tag "APOST".

CHAPTER FIVE
METHOD

5.1 Overview

Classifiers only predict a single class variable. The classification is predicting a single class variable y given a factor of features $x = \{x_1, x_2, \dots, x_k\}$. Graphical models provide to model many interdependent variables. The simplest form of dependency, in which the output variables in the graphical model are arranged in a sequence. Thus, we aim to design a model for NER task. The NER task is, for a given sentence, to segment which words are part of an entity, and to classify each entity by type (person, organization, location, and so on). The challenge of this task is that many NE strings appear even in a large set of training, so the system should only define them on the basis of content.

An approach to the NER is to classify each word independently as NE types or not. The problem with this approach assumes that all NE tags are independent, according to the input. In fact, neighboring words are dependent on named entity tags; for example, "*Mustafa Kemal"* is a person, "*Mustafa Kemal Mahallesi (Mustafa Kemal District)" is* a location. One way to loosen this assumption of independence is to arrange the output variables in a sequence or linear chain. The basic IE technique is to treat the problem as a sequence labeling problem such as POS tagging and NER.

In this chapter, we present more details about two approaches we proposed for NER system, where we used MALLET toolkit (McCallum, 2002). Before worked on MALLET, we experimented to perform HMM and CRF models using some tools. Firstly, we worked on the development of our own tools, which can apply HMM and CRF models on sequential data, are shown in Figure 5.1, 5.2 and 5.3.

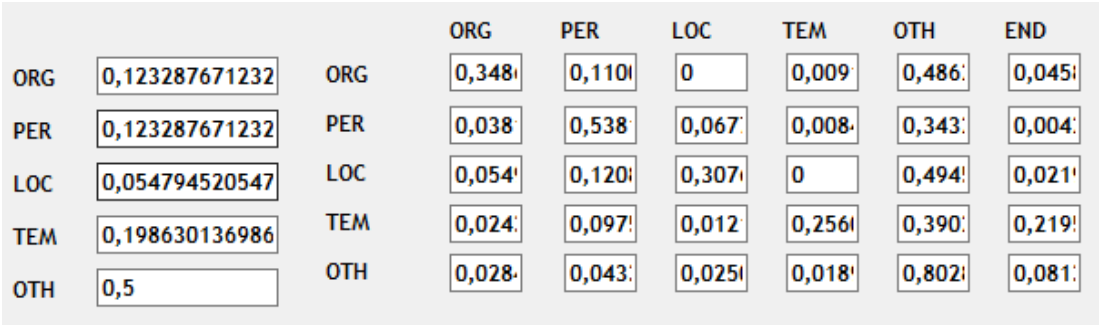

Figure 5.1 The calculation of initial probabilities in our own tool

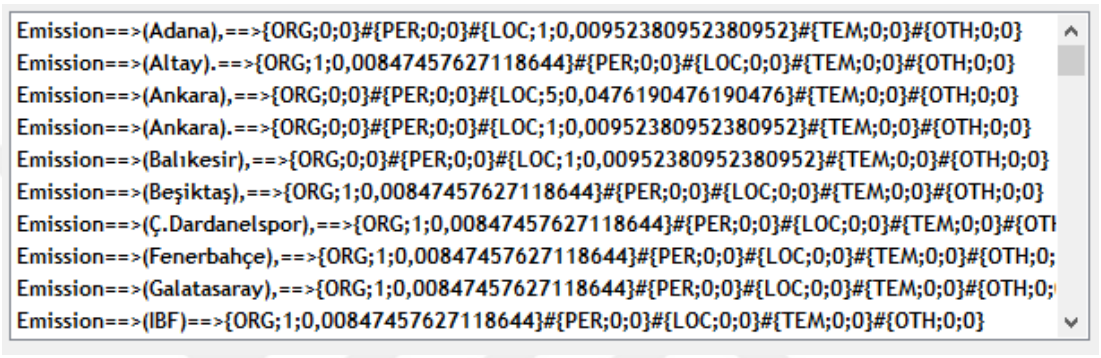

Figure 5.2 The calculation of emission probabilities in our own tool

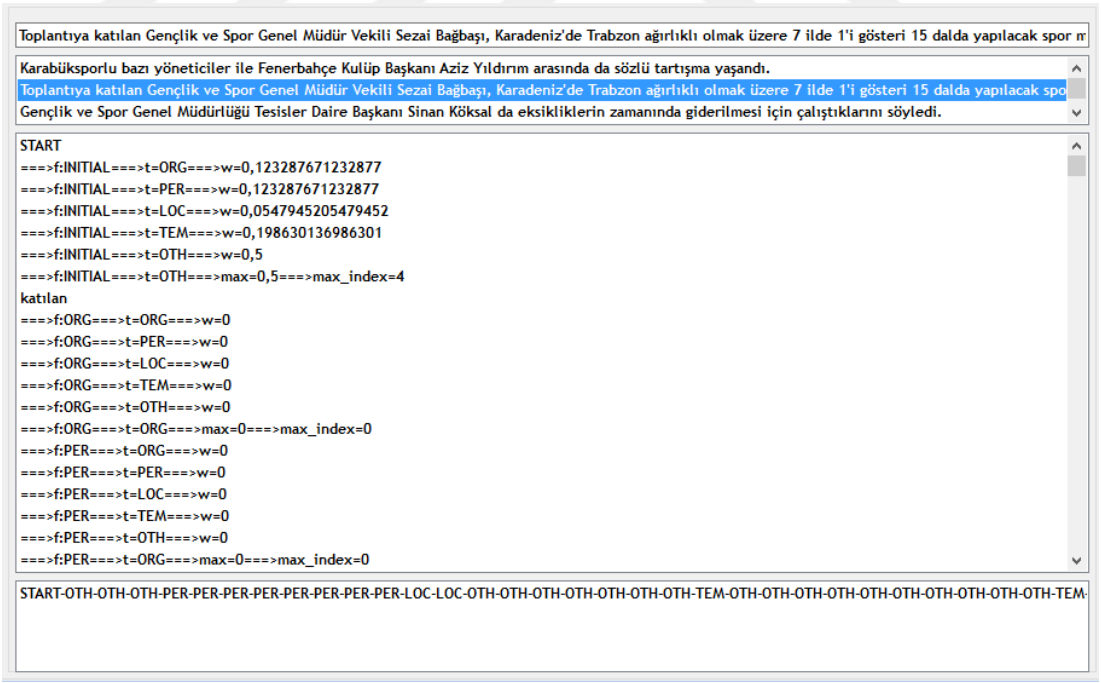

Figure 5.3 The inference step in our own tool

Then we tried to develop the models on Weka (Hall et al., 2009). We performed some preprocessing steps such as the conversion of text to a particular format as shown

in Figure 5.4. The file contains two parts. The first part contains descriptive information about the features. In data part, a line specifies each sentence and a line contains the values of features of each token in a sentence and the features are separated sequentially by a comma for each token. We used Line Feed (\n) character to separate tokens. After expressing the sequence of feature's value for tokens are existed in a same instance between double quotes, we showed the class label to the end of the line after the comma. Weka doesn't support CRF model. Then, we used R Studio (RStudio Team, 2015) due to the speed of data processing and the available libraries. We worked to implement feature extractor engine in R and the sample result for an initial matrix is shown in Figure 5.5.

```
@relation nmetu-corpus-token-disambiguated

 @attribute bag relational
   @attribute firstLetterUppercase {1,0}
   @attribute letterMixedcase {1,0}
   @attribute leastAdigit {1,0}
   @attribute finishWithPunctuation {1,0}
   @attribute containsPunctuation {1,0}
   @attribute containsApostrophe {1,0}
   @attribute getCharacterLength numeric
   @attribute existInPersonNameDict {1,0}
   @attribute existInNumberDict {1,0}
   @attribute existInDatetimeDict {1,0}
   @attribute existInCountryCityTownDict {1,0}
   @attribute resultOfOflazerResultPosTag {1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}
   @attribute containsPropTag {1,0}
   @attribute positionInSentence {1,2,3}
 @end bag
 @attribute ne_type {PER,ORG,TEM,LOC,OTH}

@data
"1,1,0,0,0,0,3,0,0,0,0,6,0,1\n0,0,0,0,0,0,8,0,0,0,0,1,0,2\n0,0,0,0,0,0,3,0,1,0,0,6,0,2\n0,0,0,1,0,0,6,0,0,0,0,1,0,3",OTH
```

Figure 5.4 The arff file for HMM classifier using rule-based features in Weka

1	0.08823529	0.03623507	0.04275574	0.03584989	0.796924

Figure 5.5 The sample results of initial matrix in R Studio.

After the examinations we have made using the tools mentioned above, we developed HMM and CRF models using MALLET. HMM was successfully applied to IE as being a sequence-labeling tool. However, HMM has difficulty modeling for non-independent features that overlap each other. For example, an HMM might specify which word x is likely for a given label y via $p(x|y)$. But often the POS tag, as well as that of the surrounding words, character *n*-grams, capitalization patterns all carry important information. HMM cannot model them easily because the generative story limits what can be generated by a state variable.

The other method is CRFs are distinctive graphical models that model these overlapping, non-independent features. A special case, the linear chain CRF, can be considered as an undirected graphical model version of HMM. It is as efficient as HMMs, where the sum-product algorithm and max-product algorithm are still applying. In a different dimension, HMMs are the sequence version of Naive Bayes models, and linear chain CRFs are the sequence version of logistic regression.

In this thesis, the basic components of the designed system closely follow as shown in Figure 5.6.

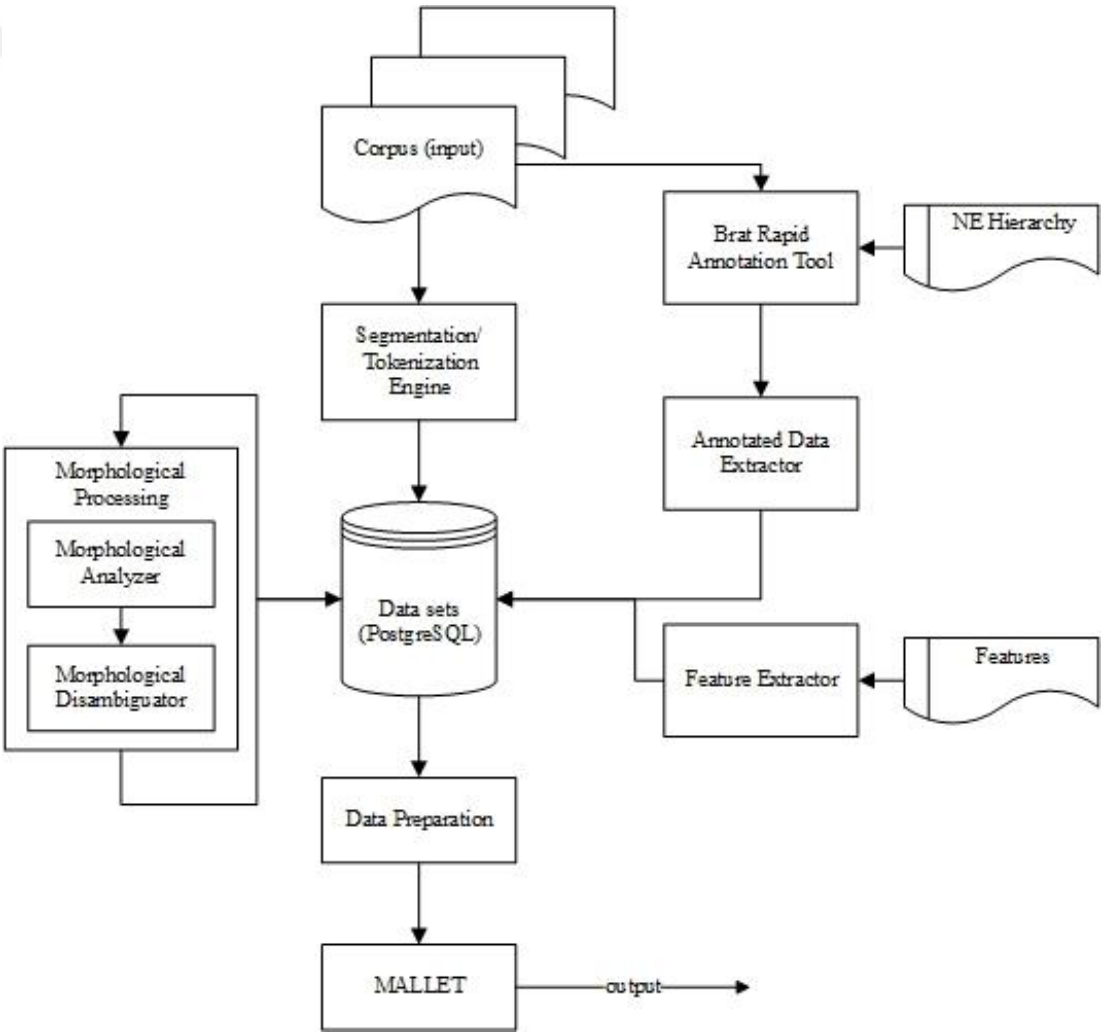

Figure 5.6 General architecture of our proposed NER system

5.2 HMM Based System

In our initial experimentations, we aimed to build HMM using multi features for each token, to apply the formulas of HMM as detailed in Section 3.3 and to perform the calculations as in the following example. An example of Hidden Markov model for a short sample sentence, which has three tokens, is shown in Figure 5.10. The steps in the HMM model for the sample sentence are shown below. We selected a sample sentence exist in the document *"00017113"* from METU-NER and its tokens and features are shown in Figure 5.7,5.8 and 5.9. We divided instances in the document as the rate of 66% training set (93 sentences) and the rest as test set (50 sentences).

- The state space:

$$S = \{s_1, s_2, s_3, s_4, s_5\}, N = 5$$
$$S = \{PER, ORG, LOC, TEM, OTH\}$$

- The selected features:

{"firstLetterUppercase","leastAdigit","containsApostrophe","getCharacterLength","positionInSentence","existInCountryCityTownDict"}

- A sample test instance/sentence (*sentence_id 3703*):

"*1951-52 arası Paris'tedir.*" *(Between 1951-52, he/she is in Paris.)*

```
1951-52 feature(revexistInCountryCityTownDict)=1.0 feature(positionB)=1.0
feature(charlen7)=1.0 feature(revcontainsApostrophe)=1.0 feature(leastAdigit)=1.0
feature(revfirstLetterUppercase)=1.0 property(revfirstLetterUppercase)=1.0
property(leastAdigit)=1.0 property(revcontainsApostrophe)=1.0 property(charlen7)=1.0
property(positionB)=1.0 property(revexistInCountryCityTownDict)=1.0
```

Figure 5.7 The features of 1st token (true NE type is TEM): "*1951-52*"

```
arası feature(revexistInCountryCityTownDict)=1.0 feature(positionI)=1.0 feature(charlen5)=1.0
feature(revcontainsApostrophe)=1.0 feature(revleastAdigit)=1.0
feature(revfirstLetterUppercase)=1.0 property(revfirstLetterUppercase)=1.0
property(revleastAdigit)=1.0 property(revcontainsApostrophe)=1.0 property(charlen5)=1.0
property(positionI)=1.0 property(revexistInCountryCityTownDict)=1.0
```

Figure 5.8 The features of 2nd token (true NE type is OTH): "*arası*"

```
Paris'tedir. feature(existInCountryCityTownDict)=1.0 feature(positionE)=1.0
feature(charlen12)=1.0 feature(containsApostrophe)=1.0 feature(revleastAdigit)=1.0
feature(firstLetterUppercase)=1.0 property(firstLetterUppercase)=1.0
property(revleastAdigit)=1.0 property(containsApostrophe)=1.0 property(charlen12)=1.0
property(positionE)=1.0 property(existInCountryCityTownDict)=1.0
```

Figure 5.9 The features of 3rd token (true NE type is LOC): "*Paris'tedir.*"

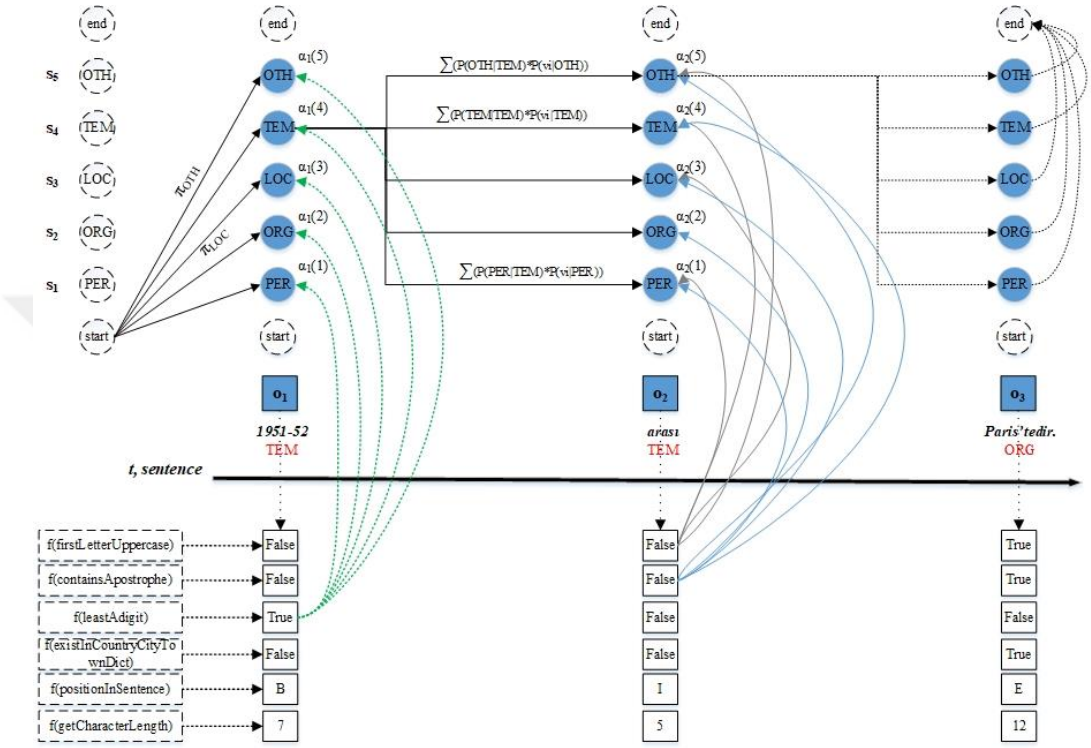

Figure 5.10 An example of our proposed HMM system

- The calculation of initial (prior) probabilities:

$$\Pi = [\pi_i]$$

$$\pi_{PER} = 0.27; \pi_{ORG} = 0.0; \pi_{LOC} = 0.04; \pi_{TEM} = 0.09; \pi_{OTH} = 0.60$$

$$\sum_{i=1}^{6} \pi_i = 1$$

$$\pi_{PER\_END} = 0.01; \pi_{ORG\_END} = 0.0; \pi_{LOC\_END} = 0.01; \pi_{TEM\_END} = 0.0; \pi_{OTH\_END} = 0.98$$

- The calculation of transition probabilities:

$A = [a_{ij}]$ where $a_{ij} \equiv P(q_{t+1} = s_j | q_t = s_i),\ 1 \le j \le N, 1 \le m \le M$

$$A = \begin{matrix} LOC \\ OTH \\ PER \\ TEM \\ ORG \end{matrix} \begin{bmatrix} 0.29 & 0.67 & 0 & 0.04 & 0 \\ 0.03 & 0.93 & 0.01 & 0.02 & 0.01 \\ 0.05 & 0.47 & 0.45 & 0.02 & 0.01 \\ 0.06 & 0.43 & 0 & 0.51 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

- The calculation of emission probabilities:

$B = [b_j(m)]$ where $b_j(m) \equiv P(O_t = v_m | q_t = s_j),\ 1 \le j \le N, 1 \le m \le M$

$$V = \{v_1, v_2, v_3, \dots, v_{28}\}, M = 28$$

V={"revexistInCountryCityTownDict","positionB","charlen7","revcontainsApostrophe","revleastAdigit","firstLetterUppercase","positionI","charlen10","revfirstLetterUppercase","charlen6","charlen8","existInCountryCityTownDict","charlen5","charlen3","positionE","charlen11","containsApostrophe","charlen9","charlen2","charlen4","leastAdigit","charlen1","charlen12","charlen15","charlen13","charlen16","charlen14","charlen17"}

$$b_{1\_LOC}(\text{existInCountryCityTownDict}) = \frac{6}{53} = 0.11$$

$$b_{2\_OTH}(\text{existInCountryCityTownDict}) = \frac{8}{1073} = 0.007$$

$$b_{3\_TEM}(\text{existInCountryCityTownDict}) = \frac{0}{48} = 0$$

$$b_{4\_PER}(\text{existInCountryCityTownDict}) = \frac{6}{93} = 0.065$$

$$b_{5\_ORG}(\text{existInCountryCityTownDict}) = \frac{1}{19} = 0.052$$

The steps taken in the MALLET toolkit are as follows, respectively. After applied this step for a sample test sentence, the output is shown in Figure 5.11. The first column shows the token, the second columns show its features and the next columns are shown correct label, predicted label and the flag (‘T’ or ’F’) of comparison result of correct and predicted labels.

```
1951-52 revexistInCountryCityTownDict positionB charlen7 revcontainsApostrophe leastAdigit
revfirstLetterUppercase TEM  TEM T
arası   revexistInCountryCityTownDict positionI charlen5 revcontainsApostrophe revleastAdigit
revfirstLetterUppercase TEM   TEM T
Paris'tedir.    existInCountryCityTownDict positionE charlen12 containsApostrophe
revleastAdigit firstLetterUppercase LOC   LOC T
```

Figure 5.11 The HMM result output of an example test sentence

1. The sentences in the documents to be used for the training and test dataset are converted to objects from the "instance" class, which is a mallet type.

```
trainingInstances.addThruPipe(new LineGroupIterator(new
   BufferedReader(new InputStreamReader((new
   FileInputStream(trainingFilename)))),
   Pattern.compile("^\\s*$"), true));
testingInstances.addThruPipe(new LineGroupIterator(new
   BufferedReader(new InputStreamReader((new
   FileInputStream(testingFilename)))),
   Pattern.compile("^\\s*$"), true));
```

2. The base structure of HMM is built.

```
HMM hmm = new HMM(pipe, null);
```

3. The states are generated using labels to create a first-order Markov model, the transitions are defined between these states and the values of initial, transition, emission and end probability matrixes are calculated.

```
hmm.addStatesForLabelsConnectedAsIn(trainingInstances);
```

4. The trainer model is built.

```
HMMTrainerByLikelihood trainer = new
   HMMTrainerByLikelihood(hmm);
```

5. The values of precision, recall and F1-score are determined on a per-class basis for training and testing instances.

```
TransducerEvaluator trainingEvaluator = new
   PerClassAccuracyEvaluator(trainingInstances,
   "training");
TransducerEvaluator testingEvaluator = new
   PerClassAccuracyEvaluator(testingInstances, "testing");
```

After the fifth step, the training process starts. This process continues until the convergence value or the number of iterations given as parameters. The convergence value is changeable and we used 0.001 as it in our experiments. As default, MALLET

has full dynamic programming implementation of the Forward-Backward "Sum-(Product)-Lattice" algorithm. The size of generated lattice is related to input size and the number of states. When the lattice is actually sparse, especially when the number of states is large and the input/sequence is long, it is not very efficient. In the first step in lattice, we take the values of initial probabilities as an alpha value. The weights are calculated and normalized for each state's transition alternatives using Forward algorithm.

```
while (pl2i.hasNext()) {
pl2i.next();
String key = pl2i.getKey();
              logEmissionProb =
             hmm.emissionMultinomial[((Alphabet)es.getAlphabe
             t()).lookupIndex(key)].
             logProbability(destIndex);
              double logTransitionProb2 =
             hmm.transitionMultinomial
             [source.getIndex()].logProbability(source.destin
             ationNames[transIndex]);
              weights[transIndex] += (logEmissionProb *
             logTransitionProb2);
}
```

The gamma values are calculated using Backward algorithm.

```
gammas[latticeLength - 1][i] = nodes[latticeLength -
                1][i].alpha + nodes[latticeLength -
                1][i].beta - totalWeight;
```

6. Next, MaxLatticeDefault method initializes the forward path or Viterbi Forward.

5.3 CRFs Based System

After our initial experimentations, we built CRF using MALLET to apply the formulas of CRF as detailed in Section 3.5 and perform the calculations as in the following example. We used the same sentence with CRF which has three tokens is used with HMM. The steps in the CRF model for the sample sentence are shown below. The tokens and features are shown in Figure 5.12,5.13 and 5.14 for window size [-1,+1]. We used the same ratio of distribution for training and test sets.

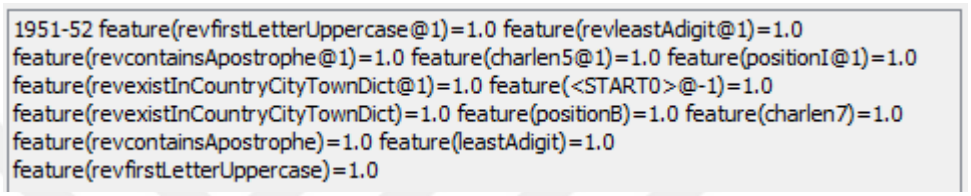

Figure 5.12 The features of 1st token (true NE type is TEM): "*1951-52*"

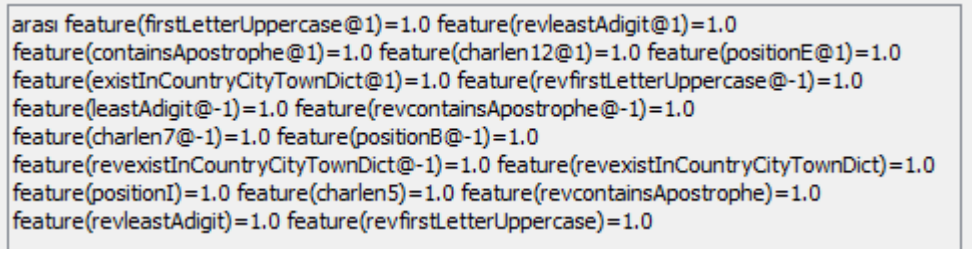

Figure 5.13 The features of 2nd token (true NE type is OTH): "*arası*"

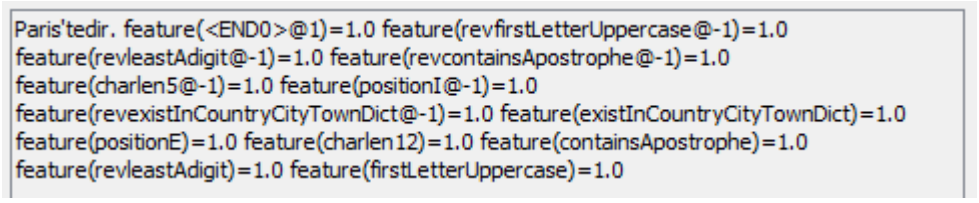

Figure 5.14 The features of 3rd token (true NE type is LOC): "*Paris'tedir.*"

The steps taken for a linear-chain CRF model in the MALLET toolkit are as follows:

1. The conjunctions matrix is created according to window size.

2. The given documents exist in training and test dataset are converted to "instance" which is mallet type.

3. The base structure of CRF is built and the default sum-product and max-product inference engines initializes. The input alphabet is shown in Figure 5.15. The output alphabet contains LOC, OTH, TEM, PER, ORG.

```
CRF crf = new CRF(trainingData.getDataAlphabet(),
   trainingData.getTargetAlphabet());
sumLatticeFactory = new SumLatticeDefault.Factory();
maxLatticeFactory = new MaxLatticeDefault.Factory();
```

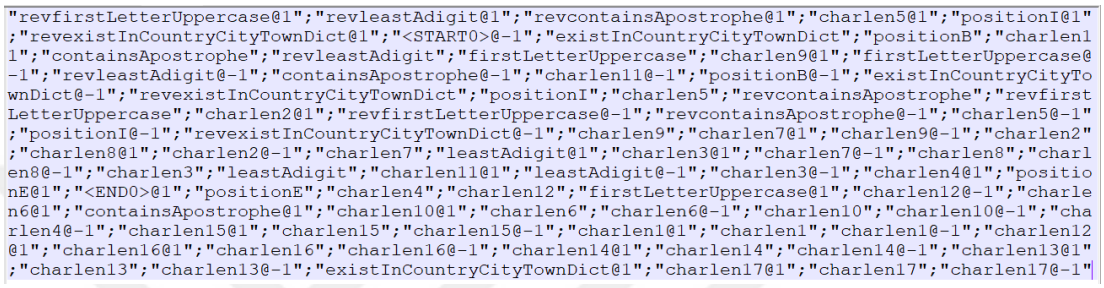

Figure 5.15 The input alphabet into CRF model for a sample sentence

4. A new empty Factors with a new empty weightsAlphabet are constructed.

5. The finite state machine is constructed.

```
crf.addFullyConnectedStatesForLabels();
```

6. The model's weights are initialized.

```
crf.setWeightsDimensionAsIn(trainingData, false);
```

7. Label likelihood model is built (8 threads were used in our experimentations)

```
CRFOptimizableByBatchLabelLikelihood batchOptLabel = new
   CRFOptimizableByBatchLabelLikelihood(crf, trainingData,
   numThreads);
optLabeld = new ThreadedOptimizable(batchOptLabel,
   trainingData, crf.getParameters().getNumFactors(), new
   CRFCacheStaleIndicator(crf));
optLabel = optLabeld;
```

8. The constraints are set by running forward-backward with the output label sequence provided.

```
gatherConstraints(ilist);
```

9. The optimization is initialized and the new threads starts. Each thread has total number of factors. The tasks create that are executed in parallel and each task looks at a batch of data. The number of instances per batch are calculated.

```
batchCachedGradient.add(new double[numFactors]);
int numBatchInstances = trainingSet.size() / numBatches;
this.createTasks();
```

10. CRF trainer is built (using L-BFGS as the optimizer)

11. All setup is done, the training process continues until the convergence value is reached.

```
crfTrainer.train(trainingData, (iterationCount == 0 ?
Integer.MAX_VALUE : iterationCount));
```

12. A transducer model is evaluated and the precision, recall and F1 scores are calculated. The predicted output for a test sentence is shown in Figure 5.16.

```
evaluator.evaluate(crfTrainer, true,
   resultFile,printDetailOutput);
Sequence trueOutput = (Sequence) instance.getTarget();
Sequence predOutput = model.transduce(input);
public Sequence transduce (Sequence input)
{
   return maxLatticeFactory.newMaxLattice(this,
        (Sequence)input).bestOutputSequence();
}
```

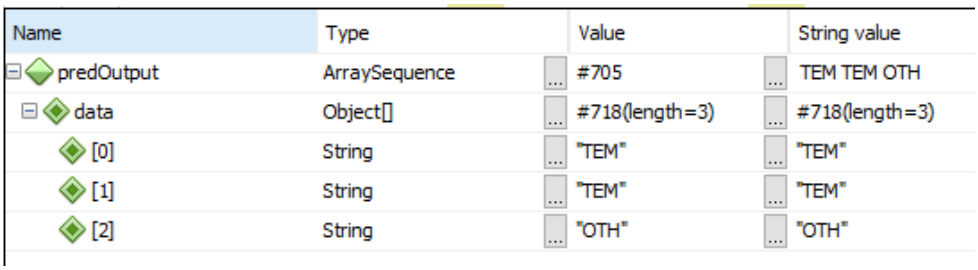

Figure 5.16 The predicted output by CRFs for an example test sentence

CHAPTER SIX
EXPERIMENTATIONS AND RESULTS

We applied on a set of experiments under different conditions in order to evaluate the performance and the behavior of the proposed NER method on using HMM and CRFs approaches. In all experiments, we used MALLET toolset, which is a Java based package for statistical natural language processing. The main objectives of the experiments are selecting the best-fitting feature set, make a comparison between HMM and CRFs approaches, behavior of CRFs model on METU-NER with different setup and evaluating the results. We also compared our results to several studies on Turkish. Moreover, we aim to investigate the impact of using features and build the widest set of features where the best results can be obtained.

Tests are performed on a system with CentOS Linux 7.5.1804 operating system, Intel(R) Xeon(R) CPU E3-1240 V2 @ 3.40GHz 8 cores and 32GB ram. As the number of fixed iterations was not determined for each experiment, the model was run until the determined convergence value was reached. We have edited MALLET toolkit on the NetBeans framework.

In this thesis, we reported our results using the commonly accepted MUCs metrics in terms of precision, recall and F-Measure and compared them with the state-of-the-art works for Turkish. Precision (P) represents the percentage of the entities that the system recognized which are actually correct, is formulated in (6.1). Recall (R) represents the percentage of the correct named entities in the text that the system identified, is formulated in (6.2). The overall accuracy result, in formula (6.3), F-Measure, is calculated by the uniformly weighted harmonic mean of precision and recall. In the below formulas, $\boldsymbol{true}_{\boldsymbol{positive}}$ corresponds to the number of items correctly recognized by the system, $\boldsymbol{false}_{\boldsymbol{positive}}$ represents the number of items not correctly recognized/predicted by the system, and $\boldsymbol{false}_{\boldsymbol{negative}}$ is the number of items existed into answer set as not retrieved by the system.

$$P = \frac{true_{positive}}{true_{positive} + false_{positive}} = \frac{\#\{correct\}}{\#\{correct + alarms\}} \quad (6.1)$$

$$R = \frac{true_{positive}}{true_{positive} + false_{negative}} = \frac{\#\{correct\}}{\#\{correct + misses\}} \quad (6.2)$$

$$F - Measure = \frac{Recall \times Precision}{\frac{1}{2} \times (Recall + Precision)} \quad (6.3)$$

Initially, for our CRF model, we have identified 4 feature categories: rule based, morphological, dictionary lookup and lexical features. These features are as follows: between RBF1 and RBF19 for rule based; LF1, LF3 and LF4 for lexical features; between DLF1 and DLF8 for dictionary lookup features and between MF1 and MF4 and MF6 for morphological features. We used these 35 features with RAW output encoding format on CRF model to measure the effects of using different windows size.

Table 6.1 The results of experiment using the feature set initially defined (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
1	[-1,+1]	536	**89.14**	**91.01**	**87.64**	**73.9**	**87.88**	**96.9**	**88.54**	**88.19**
2	[-2,+2]	248	85.33	81.39	77.73	67.75	81.16	94.53	85.47	81.39
3	[-3,+3]	60	71.57	65.5	65.3	51.04	0	85.28	77.12	67.45
4	[-4,+4]	710	88.67	89.17	86.49	65.07	85.29	94.62	87.6	86.48

As shown in Table 2.1, the current state-of-the-art systems for Turkish NER task are based on CRF and make use of language dependent features in the recent years. We decided to build our CRF model considering the base and best results that they reported in the study of Seker and Eryigit (2017) are shown in Table 6.2, ITUSYS, which we assume as the baseline results. They achieved their best results using CRF++ tool with [-3, +3] window size and IOB2 output encoding format and they used CoNLL measures for evaluation. In ITUSYS base model, they used the tokens converted to lower case in their surface form and they used morphological, lexical, gazetteer lookup and extra features in their best model.

Table 6.2 The base and best model results on ITUSYS (TRAIN7/WFS7)

Model	PER	ORG	LOC	DATE	TIME	MONEY	PERC	Overall
ITUSYSbest	91.47	94.34	89.88	89.25	91.89	100.00	98.41	92.34
ITUSYSbase	92.19	94.28	89.56	54.79	51.85	86.36	65.67	89.27

The feature used in base model corresponds to feature MF6 in our study. The equivalents of these 20 features in our study are as follows: MF1, MF3, MF4, MF5, MF6 and MF7 for morphological features; LF2, LF5 and LF6 for lexical features; between DLF9 and DLF16 for gazetteer lookup features, and RBF20, RBF21 and RBF22 for extra features. We extracted same features from tokens which are used in ITUSYS base and best models and applied them on same datasets in our system, the results are shown Table 6.3 for base model and Table 6.4 for best model. We used TRAIN7 dataset for training set and WFS7 dataset for testing. After specifying the windows size and output encoding format and determining the best-fitting features set, we have tested the same CRF model on METU-NER.

Table 6.3 Comparison the results for base model (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
5	[-1,+1]	137	76.81	**90.14**	**81.97**	**65.06**	**81.48**	**84.12**	**84.21**	**80.99**
6	[-2,+2]	233	**78.71**	87.71	79.43	62.92	81.48	79.82	82.14	79.96
7	[-3,+3]	182	77.13	87.46	78.07	63.51	81.48	78.7	83.19	79.07
8	[-4,+4]	209	77.83	85.65	75.98	58.28	76.92	75.83	82.14	77.67

Table 6.4 Comparison the results for best model (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
9	[-1,+1]	377	**92.9**	**92.72**	**90.14**	71.61	**89.23**	95.28	**90.08**	**90.24**
10	[-2,+2]	553	92.67	91.59	88.62	74.43	86.15	**95.62**	88.24	89.69
11	[-3,+3]	335	90.09	90.02	88.71	**75.05**	85.29	92.00	84.35	88.27
12	[-4,+4]	459	91.72	90.43	88.04	72.55	84.85	94.02	81.58	88.53

After these results, we decided to combine the features that we defined and the features equivalent to the used in the base study. After the combination, the newly created feature set consists of the following 45 features: between RBF1 and RBF22 for rule-based features; between LF1 and LF6 as lexical features; between DLF17 and DLF26 for dictionary lookup features and between MF1 and MF7 as morphological features. We obtained the results are shown in Table 6.5.

Table 6.5 Combination of the features identified by us and used by ITUSYS (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
13	[-1,+1]	325	91.1	**90.45**	86.18	66.8	84.85	90.49	**90.83**	**87.61**
14	[-2,+2]	273	89.95	87.59	84.76	**77.8**	**86.96**	**94.53**	86.09	87.05
15	[-3,+3]	276	89.62	87.91	84.36	76.39	84.38	92.55	85.47	86.64
16	[-4,+4]	390	**91.16**	89.76	**86.9**	65.58	84.38	87.7	85.36	87.22

When we analyzed the features that we used and the results that we obtained after the combination, we removed the repeated features from the feature set. We applied a new experiment with the remaining 40 features and obtained the results are shown in Table 6.6. As a result, we used the following features: RBF1, RBF2, RBF3 and between RBF7 and RBF22 as rule based features; LF1, LF2, LF3 and LF5 as lexical features; between DLF17 and DLF26 for dictionary lookup features and between MF1 and MF7 as morphological features.

Table 6.6 Removal the repeated equivalent features from combined feature set (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
17	[-1,+1]	21	67.07	64.73	57.07	3.03	0	0	2.63	56.18
18	[-2,+2]	204	88.67	87.79	83.72	67.58	75.41	89.8	81.58	85.03
19	[-3,+3]	528	**91.17**	**90.82**	**86.03**	**75.98**	**87.88**	**93.18**	**89.92**	**88.49**
20	[-4,+4]	202	86.41	84.54	80.57	49.6	81.82	92.37	81.08	82.11

For example, we considered as the separate features that the letters in a token consisted of upper (RF4), lower (RF5) or mixed case (RF6) letters. But in feature LF5, we handle it as a single feature. In another feature, LF4, we converted each token into a common pattern format. We thought that using feature LF4 would greatly increase the number of elements in the observation set, and thus would greatly increase the emission probability matrix. Although the feature LF6 is equivalent to the LF3 feature, the feature LF3 also has access to the fact that the token is in the middle or the end position in a sentence. Thus, we thought that a feature LF3 would have more effect on performance in NER, which is a sequence classification problem.

When we evaluate the results obtained from the experiments we have done so far, we have determined the feature set that gives the best result and the window size in

the CRF model. According to these results, [-3, +3] window size and the features which gave the best results in our CRFs model, listed in Table 6.7.

Table 6.7 The features used in experiments where the best results are obtained

Category	#	Description
Rule based Features	RBF1	Token has a uppercase letter with dot
	RBF2	Token has a uppercase letter
	RBF3	First letter is a capital in token
	RBF7	Token has only digits
	RBF8	Token has least a digit
	RBF9	Token ends with dot
	RBF10	Token ends with comma
	RBF11	Token ends with punctuation
	RBF12	Token contains punctuation
	RBF13	Token contains dash
	RBF14	Token contains slash
	RBF15	Token contains apostrophe
	RBF16	Token is in a date format
	RBF17	Token is in a clock format
	RBF18	Token is in URL format
	RBF19	Token is in e-mail address format
	RBF20	Token contains a percentage sign
	RBF21	Token contains 'saat' (*o'clock*) term
	RBF22	Token contains column indicator sign
Lexical Features	LF1	The length of token
	LF2	The pattern of token according to token contains a digit character
	LF3	The position of token in a sentence (begin, inside, end)
	LF5	The case feature of token (upper,lower,mixed,proper)
Morphological Features	MF1	The stem of word
	MF2	The POS tag
	MF3	The sequence of morphemes after POS
	MF4	Contains +Prop morphemes
	MF5	Contains some defined morphemes (is or is not in noun)
	MF6	The lower case of surface form
	MF7	The last POS tag in morphemes
Dictionary Lookup Features	Between DLF17 and DLF27	Token is existed in expanded dictionaries: first_name, last_name, abbreviations, location, person_title, generator_location, generator_organization, date_time_period, number_ordering, currency_type and organization.

Using these features listed in Table 6.7, we have done some experiments to determine the output encoding format we will get the best result. We used RAW, IO, BIO and BMEWO encoding formats. The results obtained using the RAW format were given in Table 6.6 above. For IO, BIO and BMEWO encoding formats, the

experimentation results are given in Table 6.8, Table 6.9 and Table 6.10, respectively. The results used IO format are equivalent to the results that used RAW format (in Table 6.6).

Table 6.8 The results that used IO format (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
21	[-1,+1]	21	67.07	64.73	57.07	3.03	0	0	2.63	56.18
22	[-2,+2]	204	88.67	87.79	83.72	67.58	75.41	89.8	81.58	85.03
23	[-3,+3]	528	**91.17**	**90.82**	**86.03**	**75.98**	**87.88**	**93.18**	**89.92**	**88.49**
24	[-4,+4]	202	86.41	84.54	80.57	49.6	81.82	92.37	81.08	82.11

Table 6.9 The results that used BIO format (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
19	*[-3,+3]*	*528*	*91.17*	*90.82*	*86.03*	*75.98*	*87.88*	*93.18*	*89.92*	*88.49*
25	[-1,+1]	86	83.70	81.46	75.35	59.29	74.19	83.85	81.25	79.09
26	[-2,+2]	97	84.13	80.42	75.00	61.67	70.97	82.68	82.88	78.92
27	[-3,+3]	589	**91.83**	**91.2**	**88.62**	74.17	**77.61**	**88.52**	**88.33**	**88.96**
28	[-4,+4]	391	91.08	90.19	86.05	**78.21**	75.00	87.87	80.66	87.79

Table 6.10 The results that used BMEWO format (TRAIN7/WFS7)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
29	[-1,+1]	351	91.12	88.86	86.40	66.41	84.85	93.02	89.92	87.27
30	[-2,+2]	116	83.42	81.7	76.89	63.97	67.74	87.16	82.88	79.72
31	[-3,+3]	169	86.28	85.26	80.53	63.72	48.98	89.81	84.58	82.36
32	[-4,+4]	173	85.45	83.07	78.72	85.45	70.77	87.64	82.55	81.47

According to our results, we also used the feature set and applied the output encoding format BIO in RUN27, which we obtained the best result, to the IWT and TDS test datasets, and we obtained the following results in Table 6.11 and Table 6.12.

Table 6.11 The results that used BIO format (TRAIN7/IWT)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
ITUSYS	*[-3,+3]*	*n/a*	*67.22*	*53.87*	*77.17*	*70.31*	*80.00*	*27.45*	*50.00*	*64.96*
33	[-3,+3]	533	63.84	45.02	33.97	30.30	50.00	63.77	35.56	48.37

Table 6.12 The results that used BIO format (TRAIN7/TDS)

RUN	WIN.	ITER	PER	LOC	ORG	DATE	TIME	MONEY	PERC	Overall
ITUSYS	*[-3,+3]*	*n/a*	*75.98*	*59.86*	*69.54*	*39.03*	*41.23*	*54.55*	*94.12*	*97.96*
34	[-3,+3]	589	7.19	19.56	16.41	25.74	8.70	0	0	9.22

We also applied similar experiments in METU-NER dataset that we tagged specific to our study, the results are shown in Table 6.13. In addition to previous used features, we used an organization dictionary, we called DLF27. We performed these experiments with 10-fold cross validation on the labeled 55,359 instances or sequences in METU-NER. In each fold, we tested the different set so that we would use all sequences in the test set.

Table 6.13 The results that used BIO format (METU-NER)

RUN	ITER	PER	LOC	ORG	TEM	Overall
1	78	53.29	56.43	48.70	19.74	50.24
2	285	62.28	64.17	59.97	53.21	61.16
3	207	73.31	68.10	69.04	70.80	71.22
4	312	72.43	69.65	57.83	66.67	69.50
5	214	77.17	74.71	74.01	73.00	75.28
6	329	78.62	72.61	68.74	67.61	73.27
7	338	83.48	74.92	69.44	80.31	76.58
8	47	72.99	53.55	60.49	57.28	63.65
9	97	74.01	67.81	60.49	58.23	67.67
10	283	76.52	71.31	62.82	55.17	69.60
	AVG	72.41	67.33	63.15	60.20	67.82

Next, we calculated start and final weights in MALLET using the frequency of start and final position in a sentence for each NE type. Then, we divided this frequency to the total sum of each NE type and gave the results to the lattice in model. They used into Forward and Backward algorithms. The usage of weights provided the improvement approximately 1.2% at overall results as shown in Table 6.14. MALLET accepts these weights as zero by default value.

Table 6.14 The results in BIO format using weights (METU-NER)

RUN	ITER	PER	LOC	ORG	TEM	Overall
1	239	64.28	67.70	51.34	34.52	60.47
2	176	60.62	61.32	60.63	52.69	59.77

Table 6.14 continues

3	382	74.68	71.08	68.40	71.74	72.71
4	100	63.09	65.67	53.08	48.79	61.47
5	259	78.44	78.58	73.42	73.01	76.43
6	156	76.47	69.58	68.73	63.61	71.47
7	315	82.94	73.94	69.16	78.44	75.96
8	177	81.74	71.29	69.11	73.06	74.72
9	308	75.90	73.69	64.41	66.76	71.52
10	169	74.41	68.35	60.04	52.79	66.86
	AVG	73.26	70.12	63.83	61.54	69.14

The following table, Table 6.15, shows that the results of the studies were experimented on METU Turkish Corpus. Any study has no used a common annotated dataset and same size of data into training and test phases.

Table 6.15 The results of related works on METU-NER

STUDY	**METHOD**	**PER**	**LOC**	**ORG**	**TEM**	**Overall**
Bayraktar & Temizel (2008)	Local grammar based	81.97%	n/a	n/a	n/a	n/a
Küçük & Yazıcı (2009)	Rule based	-	-	-	-	78.7%
Küçük & Yazici (2010)	Hybrid	-	-	-	-	85.95%
Küçük & Yazici (2012)	Hybrid	-	-	-	-	90.13%
Yavuz, Küçük & Yazıcı (2013)	Bayesian (I) and hybrid (II)	90.30	87.06	88.03	83.13; 80.28	(I) 87.99%; (II) 90.05%, 91.44%
Our study	CRF	**73.26**	**70.12**	**63.83**	**61.54**	**69.14**

CHAPTER SEVEN
CONCLUSIONS

The main purpose of this thesis is to develop a model for extracting named entities with sequence classifier techniques from Turkish documents. We examined HMM and CRF models as sequence classifier techniques and performed them on publically available NER datasets and a new annotated dataset METU-NER. We focused on ENAMEX (person, location and organization names) and TIMEX (date, time, period, etc.) types of named entities.

In examinations of HMM technique, we studied on presenting each token with multi features. There are many features that can be extracted from tokens in the sentence, especially in the NER studies conducted for morphologically rich languages. Next, we performed a linear-chain CRF model using MALLET toolkit to determine most effective parameters values that are used as input in it. In experiments using the CRF model, we obtained the best performance with a linear-chain CRF using the following values of parameters: [-3, +3] as window size, BIO encoding as output encoding format and the extended feature set. We performed these experiments on available NER dataset (ITU-NER) to determine their best values for these input parameters.

We generated total of 41 features in four categories: rule based, lexical based, dictionary lookup based and morphological based. When we create this extended feature set, we chose the features in our experiments with the best results are achieved. We obtained the 91.83, 91.2 and 88.62, for person names, location names and organization names respectively, in terms of F1-score.

Furthermore, in this thesis, we presented METU-NER corpus which is annotated for NER upon METU corpus. This annotated corpus contains 55,359 sentences and 628,698 tokens, in total. 37,474 NEs (53,352 tokens) were annotated with BRAT annotation tool in four categories: 15,011 person names (PER), 8,600 organization names (ORG), 10,845 location names (LOC) and 3,018 temporal expressions (TEM).

The corpus includes 25,567 annotated words containing a single token and 11,907 annotated words contain multi tokens.

Then, we evaluated our a linear-chain CRF model on METU-NER corpus with the same parameters and the same extended feature set were used in the previous dataset. In terms of F1-scores, we achieved 73.26, 70.12, 63.83 and 61.54 for person, location, organization, and temporal names, respectively. We obtained 69.14 overall F-measure score.

For future work, we will discuss to increase the weight of feature value in the CRF model for each token to be existed in the dictionary. In addition, we will apply each feature categories separately on the METU-NER corpus in order to demonstrate the impact of them on Turkish NER studies and we will discuss whether or not the values used for the number of preceding and following tokens in the window size used in the CRF model can be defined without symmetric.

REFERENCES

Adali, S., Sonmez, A. C., & Gokturk, M. (2009). An integrated architecture for processing business documents in Turkish. In *International Conference on Intelligent Text Processing and Computational Linguistics*, 394–405.

Akın, A. A., & Akın, M. D. (2007). Zemberek, an open source nlp framework for turkic languages. *Structure*, *10*, 1–5.

Alpaydin, E. (2009). *Introduction to machine learning*. MIT press.

Andale, S. (2016a). Stochastic Model / Process: Definition and Examples. Retrieved September 29, 2018 , from https://www.statisticshowto.datasciencecentral.com/stochastic-model/

Andale, S. (2016b). Deterministic: Definition and Examples. Retrieved September 29, 2018, from https://www.statisticshowto.datasciencecentral.com/deterministic/

Bayraktar, Ö., & Temizel, T. T. (2008). Person name extraction from Turkish financial news text using local grammar based approach. In *2008 23rd International Symposium on Computer and Information Sciences, ISCIS 2008*, 1–4. IEEE. doi:10.1109/ISCIS.2008.4717897

Bernal, A., Crammer, K., Hatzigeorgiou, A., & Pereira, F. (2007). Global Discriminative Learning for Higher-Accuracy Computational Gene Prediction. *PLoS Computational Biology*, *3*(3), e54. doi:10.1371/journal.pcbi.0030054

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Çelikkaya, G., Torunoğlu, D., & Eryiğit, G. (2013). Named entity recognition on real data: A preliminary investigation for Turkish. In *2013 7th International Conference on Application of Information and Communication Technologies*, 1–5. doi:10.1109/ICAICT.2013.6722801

Coding Chunkers as Taggers: IO, BIO, BMEWO, and BMEWO+ | LingPipe Blog. (2009). Retrieved September 24, 2018, from https://lingpipe-blog.com/2009/10/14/coding-chunkers-as-taggers-io-bio-bmewo-and-bmewo/

Cucerzan, S., & Yarowsky, D. (1997). Language Independent Named Entity

Recognition Combining Morphological and Contextual Evidence. *Emnlp*, 90–99. Retrieved September 18, 2018, from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.3448

Cunningham, H. (2005). Information extraction, automatic. *Encyclopedia of language and linguistics*, 665–677.

Dalkılıç, F. E., Gelişli, S., & Diri, B. (2010). Named Entity Recognition from Turkish texts. In *2010 IEEE 18th Signal Processing and Communications Applications Conference*, 918–920. doi:10.1109/SIU.2010.5653553

Demir, H., & Ozgur, A. (2014). Improving named entity recognition for morphologically rich languages using word embeddings. In *Proceedings - 2014 13th International Conference on Machine Learning and Applications, ICMLA 2014*, 117–122. IEEE. doi:10.1109/ICMLA.2014.24

Eken, B., & Tantug, C. (2015). Recognizing named entities in Turkish tweets. *Proceedings of the Fourth International Conference on Software Engineering and Applications, Dubai, UAE*.

Emekligil, E., Arslan, S., & Agin, O. (2016). A Bank Information Extraction System Based on Named Entity Recognition with CRFs from Noisy Customer Order Texts in Turkish. A.-C. Ngonga Ngomo, & P. K\vremen (Ed.), In *Knowledge Engineering and Semantic Web*, 93–102. Cham: Springer International Publishing.

Eryiğit, G. (2007). ITU Treebank Annotation Tool. In *Proceedings of the Linguistic Annotation Workshop*, LAW '07, 117–120. Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved September 24, 2018, from http://dl.acm.org/citation.cfm?id=1642059.1642078.

Eryiğit, G. (2014). ITU Turkish NLP web service. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, 1–4.

Grishman, R. (2003). Information extraction. *The Oxford handbook of computational linguistics*, 545–559.

Grishman, R., & Sundheim, B. (1996). Message Understanding Conference-6: A Brief

History. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, 466–471. Stroudsburg, PA, USA: Association for Computational Linguistics. doi:10.3115/992628.992709

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. SIGKDD Explor. Newsl., 11(1), 10–18. doi:10.1145/1656274.1656278

Hankamer, J. (1989). Lexical Representation and Process. W. Marslen-Wilson (Ed.), 392–408. Cambridge, MA, USA: MIT Press. Retrieved September 23, 2018, from http://dl.acm.org/citation.cfm?id=86149.86174.

Jurafsky, D., & Martin, J. H. (2014). *Speech and language processing* (C. 3). Pearson London.

Kay, S. M. (2006). *Intuitive Probability and Random Processes Using MATLAB®*. Boston, MA: Springer US. doi:10.1007/b104645

Kazkilinc, S., & Adali, E. (2013). Labeling Turkish news stories with CRF. In *2013 7th International Conference on Application of Information and Communication Technologies*, 1–5. IEEE. doi:10.1109/ICAICT.2013.6722634

Klinger, R., & Tomanek, K. (2007). *Classical Probabilistic Models and Conditional Random Fields*. Algorithm Engineering report. TU, Algorithm Engineering. Retrieved January 26, 2016, from https://books.google.com.tr/books?id=TBvlZwEACAAJ.

Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. Adaptive computation and machine learning. MIT Press. Retrieved January 26, 2016, from https://books.google.com.tr/books?id=7dzpHCHzNQ4C.

Kschischang, F. R., Frey, B. J., & Loeliger, H.-A. (2006). Factor Graphs and the Sum-product Algorithm. *IEEE Trans. Inf. Theor.*, *47*(2), 498–519. doi:10.1109/18.910572

Küçük, D., Arıcı, N., & Küçük, D. (2017). Named Entity Recognition in Turkish: Approaches and Issues. F. Frasincar, A. Ittoo, L. M. Nguyen, & E. Métais (Ed.), In *Natural Language Processing and Information Systems*, 176–181. Cham: Springer

International Publishing.

Küçük, D., Jacquet, G., & Steinberger, R. (2014). Named Entity Recognition on Turkish Tweets. In *LREC*.

Küçük, D., Küçük, D., & Arıcı, N. (2016). A named entity recognition dataset for Turkish. In *2016 24th Signal Processing and Communication Application Conference (SIU)*, 329–332. doi:10.1109/SIU.2016.7495744

Küçük, D., & Yazici, A. (2010). A Hybrid Named Entity Recognizer for Turkish with Applications to Different Text Genres. E. Gelenbe, R. Lent, G. Sakellari, A. Sacan, H. Toroslu, & A. Yazici (Ed.), In *Computer and Information Sciences*, 113–116. Dordrecht: Springer Netherlands.

Küçük, D., & Yazici, A. (2012). A hybrid named entity recognizer for Turkish. *Expert Systems with Applications*, *39*(3), 2733–2742. doi:10.1016/j.eswa.2011.08.131

Küçük, D., & Yazıcı, A. (2009). Named entity recognition experiments on turkish texts. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*(C. 5822 LNAI), 524–535. Springer, Berlin, Heidelberg. doi:10.1007/978-3-642-04957-6_45

Küçük, D., & Yazıcı, A. (2011). Exploiting Information Extraction Techniques for Automatic Semantic Video Indexing with an Application to Turkish News Videos. *Know.-Based Syst.*, *24*(6), 844–857. doi:10.1016/j.knosys.2011.03.006

Küçük, D., & Yazıcı, A. (2013). A Semi-automatic Text based Semantic Video Annotation System for Turkish Facilitating Multilingual Retrieval. *Expert Syst. Appl.*, *40*(9), 3398–3411. doi:10.1016/j.eswa.2012.12.048

Kudo, T. (2005). CRF++: Yet another CRF toolkit. *Software available at http://crfpp. sourceforge. net*. Retrieved September 21, 2018, from http://taku910.github.io/crfpp/.

Lafferty, J., McCallum, A., & Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning*, *8*(June), 282–289. doi:10.1038/nprot.2006.61

Leon-Garcia, A. (2011). *Probability, Statistics, and Random Processes For Electrical Engineering*. Pearson/Prentice Hall. Retrieved September 29, 2018, from https://books.google.com.tr/books?id=-aYuAAAAQBAJ.

Li, S. Z. (2001). *Markov Random Field Modeling in Image Analysis*. Berlin, Heidelberg: Springer-Verlag.

Mansouri, A., Affendey, L. S., & Mamat, A. (2008). Named Entity Recognition Approaches.

McCallum, A. K. (2002). MALLET: A Machine Learning for Language Toolkit. Retrieved September, 21, 2018, from http://mallet.cs.umass.edu.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in, &ctor Space. Retrieved from http://arxiv.org/abs/1301.3781.

Mosallam, Y., Abi-Haidar, A., & Ganascia, J.-G. (2014). Unsupervised Named Entity Recognition and Disambiguation: An Application to Old French Journals. P. Perner (Ed.), In *Advances in Data Mining. Applications and Theoretical Aspects*, 12–23. Cham: Springer International Publishing.

MUC-6. (n.d.). Retrieved September 21, 2018, from https://cs.nyu.edu/faculty/grishman/muc6.html.

Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Linguisticae Investigationes*, *30*(1), 3–26. Retrieved September 21, 2018, from http://www.ingentaconnect.com/content/jbp/li/2007/00000030/00000001/art00002.

Oflazer, K. (1994). Two-level description of Turkish morphology. *Literary and linguistic computing*, *9*(2), 137–148.

Ozdemir, C. B., & Diri, B. (2011). Web-Ear, an information retrieval system that uses reported speech expressions in Turkish. In *2011 International Symposium on Innovations in Intelligent Systems and Applications*, 370–374. IEEE. doi:10.1109/INISTA.2011.5946094

Ozkaya, S., & Diri, B. (2011). Named Entity Recognition by Conditional Random

Fields from Turkish informal texts. In *2011 IEEE 19th Signal Processing and Communications Applications Conference (SIU)*, 662–665. IEEE. doi:10.1109/SIU.2011.5929737

Öztürkmenoğlu, O. (2012). *Design and Implementation of Turkish Question Answering System*. MSc Thesis, Dokuz Eylül University, İzmir.

Pamay, T., Sulubacak, U., Torunoğlu-Selamet, D., & Eryiğit, G. (2015). The Annotation Process of the ITU Web Treebank. In *Proceedings of The 9th Linguistic Annotation Workshop*, 95–101. Association for Computational Linguistics. doi:10.3115/v1/W15-1610

Poibeau, T., & Kosseim, L. (2001). Proper Name Extraction from Non-Journalistic Texts. In *Computational Linguistics in the Netherlands*. Leiden, The Netherlands: Brill.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286. doi:10.1109/5.18626

RStudio Team. (2015). RStudio: Integrated Development Environment for R. Boston, MA. Retrieved September 23, 2018, from http://www.rstudio.com/

Sahin, M., Sulubacak, U., & Eryigit, G. (2013). Redefinition of Turkish morphology using flag diacritics. In, *Proceedings of The Tenth Symposium on Natural Language Processing (SNLP-2013)*. Phuket, Thailand.

Sak, H., Güngör, T., & Saraçlar, M. (2008). Turkish Language Resources: Morphological Parser, Morphological Disambiguator and Web Corpus. B. Nordström, & A. Ranta (Ed.), In *Advances in Natural Language Processing*, 417–427. Berlin, Heidelberg: Springer Berlin Heidelberg.

Sarawagi, S. (n.d.). CRF Project Page. Retrieved September 23, 2018, from http://crf.sourceforge.net/.

Say, B., Zeyrek, D., Oflazer, K., & Özge, U. (2002). Development of a corpus and a treebank for present-day written Turkish. In *Proceedings of the eleventh international conference of Turkish linguistics*, 183–192.

Seker, G. A., & Eryigit, G. (2012). Initial Explorations on using CRFs for Turkish Named Entity Recognition. In *Proceedings of COLING 2012*, 2459–2474.

Seker, G. A., & Eryigit, G. (2017). Extending a CRF based named entity recognition model for Turkish well formed text and user generated content. *Semantic Web*, *8*, 625–642.

Stenetorp, P., Pyysalo, S., & Topi, G. (2012). BRAT : a Web based Tool for NLP-Assisted Text Annotation. *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, 102–107. Retrieved September 23, 2018, from https://dl.acm.org/citation.cfm?id=2380942.

Stern, D. H., Graepel, T., & MacKay, D. J. C. (2004). Modelling Uncertainty in the Game of Go. In *NIPS*.

Stonebraker, M., Rowe, L. A., & Hirohama, M. (1990). The Implementation of POSTGRES. *IEEE Trans. on Knowl. and Data Eng.*, *2*(1), 125–142. doi:10.1109/69.50912

Sutton, C., & McCallum, A. (2010). An Introduction to Conditional Random Fields. Retrieved October 4, 2018, from http://arxiv.org/abs/1011.4088.

Taskar, B., Klein, D., Collins, M., Koller, D., & Manning, C. (2004). Max-margin parsing. *In Proceedings of EMNLP*.

Tatar, S., & Cicekli, I. (2011). Automatic rule learning exploiting morphological features for named entity recognition in Turkish. *Journal of Information Science*, *37*(2), 137–151. doi:10.1177/0165551511398573

Tür, G., Hakkani-Tür, D., & Oflazer, K. (2003). A statistical information extraction system for Turkish. *Natural Language Engineering*, *9*(2), 181–210. doi:10.1017/S135132490200284X

Turmo, J., Ageno, A., & Català, N. (2006). Adaptive Information Extraction. *ACM Comput. Surv.*, *38*(2). doi:10.1145/1132956.1132957

Weisstein, E. W. (n.d.). Bipartite Graph. Retrieved October 4, 2018, from http://mathworld.wolfram.com/BipartiteGraph.html.

Yavuz, S. R., Küçük, D. ve Yazıcı, A. (2013). Named Entity Recognition in Turkish with Bayesian Learning and Hybrid Approaches. E. Gelenbe ve R. Lent (Ed.), In *Information Sciences and Systems 2013* (129–138). Cham: Springer International Publishing.

Yeniterzi, R. (2011). Exploiting Morphology in Turkish Named Entity Recognition System. In *Proceedings of the ACL 2011 Student Session*, HLT-SS '11, 105–110. Stroudsburg, PA, USA: Association for Computational Linguistics. Retrieved September 18, 2018, from http://dl.acm.org/citation.cfm?id=2000976.2000995.

APPENDICES

Appendix-1: The Extending Feature Set

Category	FeatureId	Description
Rule based Features	RBF1	Token has a uppercase letter with dot
	RBF2	Token has a uppercase letter
	RBF3	First letter is a capital in token
	RBF7	Token has only digits
	RBF8	Token has least a digit
	RBF9	Token ends with dot
	RBF10	Token ends with comma
	RBF11	Token ends with punctuation
	RBF12	Token contains punctuation
	RBF13	Token contains dash
	RBF14	Token contains slash
	RBF15	Token contains apostrophe
	RBF16	Token is in a date format
	RBF17	Token is in a clock format
	RBF18	Token is in URL format
	RBF19	Token is in e-mail address format
	RBF20	Token contains a percentage sign
	RBF21	Token contains 'saat' (*o'clock*) term
	RBF22	Token contains column indicator sign
Lexical Features	LF1	The length of token
	LF2	The pattern of token according to token contains a digit character
	LF3	The position of token in a sentence (begin, inside, end)
	LF4	The pattern of surface form of token
	LF5	The case feature of token (upper,lower,mixed,proper)
	LF6	The position of token in a sentence (begin or not)
Morphological Features	MF1	The stem of word
	MF2	The POS tag
	MF3	The sequence of morphemes after POS
	MF4	Contains +Prop morphemes
	MF5	Contains some defined morphemes (is or is not in noun)
	MF6	The lower case of surface form
	MF7	The last POS tag in morphemes
Dictionary Lookup Features	Between DLF17 and DLF27	Token is existed in expanded dictionaries: first_name, last_name, abbreviations, location, person_title, generator_location, generator_organization, date_time_period, number_ordering, currency_type and organization.

#	Author	Publisher	Dataset	NE Type	Approach	Features	Evaluation	Best-results (F-measure)
1	Cucerzan and Yarowsky (1999)	SIGDAT Conference on Empirical Methods	5207 words in text; 150 words in training seed	First name, Last name, City/Country	EM-style bootstrapping	word internal and contextual information	MUC	53.04
2	Tür, Hakkani-Tür & Oflazer (2003)	Natural Language Engineering	Milliyet newspaper articles	ENAMEX	HMM	lexical, contextual, morphological	MUC	91.56 (PER:92.63; LOC:95.33; ORG:89.77)
3	Bayraktar and Temizel (2008)	International Symposium on Computer and Information Sciences	Economy Corpus and METU Corpus	Person	Local grammar based	concordance analysis, collocation analysis, pattern generation	MUC	81.97
4	Küçük and Yazıcı (2009)	International Conference on Flexible Query Answering Systems	METU Corpus (I), child stories (II) and historical texts (III)	ENAMEX, NUMEX, TIMEX	Rule-based	lexical resources and pattern bases	MUC	(I) 78.7; (II) 69.3; (III) 55.3
5	Adali, Sonmez ve Gokturk (2009)	International Conference on Intelligent Text Processing and Computational Linguistics	500 (I),1000 (II) and 2000 (III) financial documents	Date, time, fax number, currency type, customer name, account number etc.	Rule-based and dictionary-based	Rule-based and dictionary-based	MUC	(I) 97; (II) 98; (III) 99
6	Dalkılıç, Gelişli & Diri (2010)	Signal Processing and Communications Applications Conference	10 documents in politics (I), economy (II) and health (III)	ENAMEX	Rule-based	Case feature, the rules of class, small size gazetteers	MUC	(I) PER:0.78; LOC:0.84; ORG:0.73 (II) PER:0.88; LOC:0.90; ORG:0.82 (III) PER:0.79; LOC:0.90; ORG:0.85
7	Dilek Küçük and Yazici (2010)	International Symposium on Computer and Information Sciences	METU Corpus (I), financial news texts (II), child stories (III) and historical texts (IV)	ENAMEX, NUMEX, TIMEX	Hybrid	lexical resources and pattern bases+rote learning	MUC	(I) 85.95;(II) 74.23;(III) 85.06;(IV) 66.96
8	Ozkaya and Diri (2011)	Signal Processing and Communications Applications (SIU)	150 e-mails in academic (I), business (II), personal (III)	ENAMEX	CRF (CRF by Sarawagi)	Token, token length, frequency, isAbbreviation, POS-tag, firstWordInSentence, case, containsPunctuation, existInDictionaries, existInSubject	MUC	(I) PER:0.94 LOC:0.68 ORG:0.83 (II) PER:0.97 LOC:0.79 ORG:0.84 (III) PER:0.92 LOC:0.70 ORG:0.90
9	Tatar and Cicekli (2011)	Journal of Information Science	TurkIE	ENAMEX,TIMEX	Automatic rule learning	orthographical, contextual, lexical and morphological features	MUC	91.08 (PER:94.33 LOC:90.03 ORG:87.68 DATE:96.50 TIME 92:05)
10	Yeniterzi (2011)	ACL 2011 Student Session	Milliyet newspaper articles	ENAMEX	CRF++	root, POS-tag, morphemes, containsPropTag,case	CoNLL	88.94 (PER:89.32 ORG:83.50 LOC:92.15)

#	Author	Publisher	Dataset	NE Type	Approach	Features	Evaluation	Best-results (F-measure)
11	Dilek Küçük and Yazici (2012)	Expert Systems with Applications	METU Corpus (I), financial news texts (II), child stories (III) and historical texts (IV)	ENAMEX, NUMEX, TIMEX	Hybrid	lexical resources and pattern bases+rote learning+capitalization	OTHER	(I) 90.13; (II) 76.80; (III) 92.47; (IV) 80.66
12	Seker and Eryigit (2012)	COLING	Milliyet newspaper articles	ENAMEX	CRF++ (window size [-3,+3], IOB2 output format)	morphological features (stem,POS,nounCase,prop,morphemes),lexical (case,startOfSent) and gazetteers	MUC	94.59 (PER:94.81 ORG:91.09 LOC:93.35)
							CoNLL	91.94 (PER:92.94 ORG:88.77 LOC:92.93)
13	Kazkilinc and Adali (2013)	International Conference on Application of Information and Communication Technologies	75 documents for each title (Turkey, World,Politics, Economy, Science and Technology and Sports)	Subject, Verb, Location, Time	CRF	rule based, morphological, syntactic and contextual	MUC	SUB:0.72 VERB:0.684 LOC:0.695 TIME:0.55
14	Çelikkaya, Torunoğlu & Eryiğit (2013)	International Conference on Application of Information and Communication Technologies	Milliyet newspaper articles (I), Forum (II), Speech-to-Text (III), Twitter (IV)	ENAMEX	CRF++ (window size [-3,+3])	word,stem,POS tag,nounCase,propNoun,morphemes, case,firstWord,gazetteers	CoNLL	(I) 91.64; (II) 19.28; (III) 50.84; (IV) 5.62
15	Yavuz, Küçük & Yazıcı (2013)	Information Sciences and Systems 2013	METUCorpus	ENAMEX, NUMEX, TIMEX	Bayesian Learning (I) and hybrid (rule and Bayesian) (II)	case feature, length, alphanumeric, nymble features, lexical resource feature	OTHER	(I) 87.99 (PER:90.30; LOC:87.06; ORG:88.03; DATE:83.13; TIME:80.28; MON:84.73; PERC:92.01) (II) 90.05; 91.44
16	Demir and Ozgur (2014)	International Conference on Machine Learning and Applications	Milliyet newspaper articles, data collected from several Turkish news sites	ENAMEX	semi-supervised (neural networks, BIEWO format)	tokens in [-2,2] window size, previous three tag, capitalized, dijit, punctuation, prefixes,suffixes, word2vec clusters, etc.	CoNLL	91.85 (PER:94.69; ORG:85.78; LOC:92.40)
17	Eken and Tantug (2015)	International Conference on Software Engineering and Applications	Milliyet newspaper articles, Tweet data set 1, Tweet data set 2	ENAMEX, NUMEX, TIMEX	CRF++ (IOB2 output format)	stem, morphemes, POS tag, nounCase, propTag, firstFour, lastFour, containsApost, gazetteer	CoNLL	63.77 on TweetsTestSet1
18	Emekligil, Arslan & Agin (2016)	Knowledge Engineering and Semantic Web	Bank Customer Transaction Orders	ACC,AMOUNT,CLIENT,CURR,EXPL,IBAN,ORG	CRF (MALLET, windows size [-3,+3])	punctuation,wordshapes, names, n-grams	CoNLL	72.77 (ACC:86.33; AMOUNT:72.23; CLIENT:36.73; CURR:86:51; EXPL:21.05; IBAN:86.92; ORG:67.18)
19	Seker ve Eryigit (2017)	Semantic Web	Milliyet newspaper articles (I), Forum (II), Twitter (III)	ENAMEX, NUMEX, TIMEX	CRF++ (window size [-3,+3], IOB2 output format)	morphological features (stem,POS,nounCase,prop,morphemes),lexical (case,startOfSent), gazetteers,numericValue,percentageSign,oclockTerm,columnIndicator	CoNLL	(I) 92.34 (PER:91.47 ORG:94.34 LOC:89.88 DATE:89.25 TIME:91.89 MONEY:100 PERCENT:98.41)
								(II) 64.96 (PER:67.22 ORG:77.17 LOC:53.87 DATE:70.31 TIME:80.00 MONEY:27.45 PERCENT:50.00)
								(III) 67.96 (PER:75.98 ORG:69.54 LOC:59.86 DATE:39.03 TIME:41.23 MONEY:54.55 PERCENT:94.12)

Appendix-3: Precision, Recall, and F-measure Values (TRAIN7/WFS7)

run_id	PERSON			LOCATION			ORGANIZATION			OVERALL WITHOUT OTH		
	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure
1	0.8659	0.9185	**0.8914**	0.9289	0.8921	**0.9101**	0.9303	0.8285	**0.8764**	0.889	0.875	**0.8819**
2	0.845	0.8616	0.8533	0.8443	0.7855	0.8139	0.8281	0.7324	0.7773	0.8371	0.792	0.8139
3	0.7019	0.7301	0.7157	0.6744	0.6366	0.655	0.6468	0.6594	0.653	0.6873	0.6622	0.6745
4	0.8712	0.9028	0.8867	0.9188	0.8661	0.8917	0.9228	0.8139	0.8649	0.8828	0.8474	0.8648
5	0.8649	0.6909	0.7681	0.9599	0.8497	**0.9014**	0.9388	0.7275	**0.8197**	0.8975	0.7378	**0.8099**
6	0.9077	0.6948	**0.7871**	0.9378	0.8238	0.8771	0.9064	0.7068	0.7943	0.8981	0.7205	0.7996
7	0.8794	0.6869	0.7713	0.9375	0.8197	0.8746	0.9207	0.6776	0.7807	0.8937	0.709	0.7907
8	0.9061	0.682	0.7783	0.9339	0.791	0.8565	0.9174	0.6484	0.7598	0.8964	0.6853	0.7767
9	0.9268	0.9313	**0.929**	0.9409	0.9139	**0.9272**	0.9256	0.8783	**0.9014**	0.91	0.8949	**0.9024**
10	0.9359	0.9176	0.9267	0.9248	0.9071	0.9159	0.9224	0.8528	0.8862	0.9136	0.8808	0.8969
11	0.905	0.897	0.9009	0.9201	0.8811	0.9002	0.9104	0.865	0.8871	0.9001	0.866	0.8827
12	0.9209	0.9136	0.9172	0.9183	0.8907	0.9043	0.9071	0.8552	0.8804	0.9041	0.8673	0.8853
13	0.9083	0.9136	0.911	0.9306	0.8798	**0.9045**	0.8624	0.8613	0.8618	0.8865	0.866	**0.8761**
14	0.9031	0.896	0.8995	0.8844	0.8675	0.8759	0.8827	0.8151	0.8476	0.8872	0.8545	0.8705
15	*0.8984*	*0.894*	*0.8962*	*0.8999*	*0.8593*	*0.8791*	*0.8686*	*0.82*	*0.8436*	*0.8813*	*0.8519*	*0.8664*
16	0.9125	0.9107	**0.9116**	0.9161	0.8798	0.8976	0.9152	0.8273	**0.869**	0.8968	0.849	0.8722
17	0.6865	0.6555	0.6707	0.7336	0.5792	0.6473	0.4929	0.6776	0.5707	0.597	0.5304	0.5618
18	0.8942	0.8793	0.8867	0.9193	0.8402	0.8779	0.8525	0.8224	0.8372	0.8733	0.8285	0.8503
19	***0.9068***	***0.9166***	***0.9117***	***0.9401***	***0.8784***	***0.9082***	***0.8824***	***0.8394***	***0.8603***	***0.8988***	***0.8715***	***0.8849***
20	0.8637	0.8646	0.8641	0.8622	0.8292	0.8454	0.8615	0.7567	0.8057	0.8635	0.7827	0.8211
21	0.6865	0.6555	0.6707	0.7336	0.5792	0.6473	0.4929	0.6776	0.5707	0.597	0.5304	0.5618
22	0.8942	0.8793	0.8867	0.9193	0.8402	0.8779	0.8525	0.8224	0.8372	0.8733	0.8285	0.8503
23	*0.9068*	*0.9166*	***0.9117***	*0.9401*	*0.8784*	***0.9082***	*0.8824*	*0.8394*	***0.8603***	*0.8988*	*0.8715*	***0.8849***
24	0.8637	0.8646	0.8641	0.8622	0.8292	0.8454	0.8615	0.7567	0.8057	0.8635	0.7827	0.8211
25	0.8227	0.8518	0.8370	0.8350	0.7951	0.8146	0.7799	0.7287	0.7535	0.8145	0.7686	0.7909
26	0.8425	0.8400	0.8413	0.8238	0.7855	0.8042	0.7753	0.7263	0.7500	0.8150	0.7651	0.7892
27	0.9048	0.9323	**0.9183**	0.9329	0.8921	**0.912**	0.9281	0.8479	**0.8862**	0.9043	0.8753	**0.8896**
28	0.9099	0.9117	0.9108	0.9328	0.873	0.9019	0.927	0.8029	0.8605	0.9062	0.8513	0.8779
29	0.9059	0.9166	0.9112	0.9063	0.8716	**0.8886**	0.8999	0.8309	**0.8640**	0.8866	0.8593	**0.8727**
30	0.8266	0.842	0.8342	0.8326	0.8019	0.817	0.7855	0.753	0.7689	0.8105	0.7843	0.7972
31	0.8497	0.8763	**0.8628**	0.8949	0.8142	0.8526	0.8297	0.7822	0.8053	0.8420	0.8061	0.8236
32	0.8532	0.8557	0.8545	0.8478	0.8142	0.8307	0.8132	0.7628	0.7872	0.8293	0.8006	0.8147

Appendix-3 continues

run_id	DATE			TIME			MONEY			PERCENT			OVERALL WITHOUT OTH		
	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure
1	0.7321	0.7462	**0.739**	0.8529	0.9062	**0.8788**	0.9843	0.9542	0.969	0.8682	**0.9032**	0.8854	0.889	0.875	**0.8819**
2	0.7167	0.6423	0.6775	0.7568	0.875	0.8116	0.968	0.9237	0.9453	0.9091	0.8065	0.8547	0.8371	0.792	0.8139
3	0.6433	0.4231	0.5104	1	0	0	0.8433	0.8626	0.8528	0.8125	0.7339	0.7712	0.6873	0.6622	0.6745
4	0.6763	0.6269	0.6507	0.8056	0.9062	0.8529	0.9535	0.9389	0.9462	0.8983	0.8548	0.876	0.8828	0.8474	0.8648
5	0.6807	0.6231	**0.6506**	1	0.6875	**0.8148**	0.9608	0.7481	**0.8412**	0.9231	0.7742	**0.8421**	0.8975	0.7378	**0.8099**
6	0.6864	0.5808	0.6292	1	0.6875	**0.8148**	0.9381	0.6947	0.7982	0.92	0.7419	0.8214	0.8981	0.7205	0.7996
7	0.6844	0.5923	0.6351	1	0.6875	0.8148	1	0.6489	0.787	0.9216	0.7581	0.8319	0.8937	0.709	0.7907
8	0.6406	0.5346	0.5828	1	0.625	0.7692	1	0.6107	0.7583	0.92	0.7419	0.8214	0.8964	0.6853	0.7767
9	0.6918	0.7423	0.7161	0.8788	0.9062	**0.8923**	0.9837	0.9237	0.9528	0.9237	0.879	**0.9008**	0.91	0.8949	**0.9024**
10	0.7386	0.75	0.7443	0.8485	0.875	0.8615	1	0.916	0.9562	0.9211	0.8468	0.8824	0.9136	0.8808	0.8969
11	0.7671	0.7346	0.7505	0.8056	0.9062	0.8529	0.9664	0.8779	0.92	0.9151	0.7823	0.8435	0.9001	0.866	0.8827
12	0.7573	0.6962	0.7255	0.8235	0.875	0.8485	0.9833	0.9008	0.9402	0.8942	0.75	0.8158	0.9041	0.8673	0.8853
13	0.7149	0.6269	0.668	0.8235	0.875	0.8485	0.9015	0.9084	0.9049	0.9397	0.879	**0.9083**	0.8865	0.866	**0.8761**
14	0.7952	0.7615	**0.778**	0.8108	0.9375	**0.8696**	0.968	0.9237	**0.9453**	0.934	0.7984	0.8609	0.8872	0.8545	0.8705
15	*0.7625*	*0.7654*	*0.7639*	*0.8438*	*0.8438*	*0.8438*	*0.9516*	*0.9008*	*0.9255*	*0.9091*	*0.8065*	*0.8547*	*0.8813*	*0.8519*	*0.8664*
16	0.697	0.6192	0.6558	0.8438	0.8438	0.8438	0.9469	0.8168	0.877	0.887	0.8226	0.8536	0.8968	0.849	0.8722
17	1	0.0154	0.0303	1	0	0	0	0	0	0.0714	0.0161	0.0263	0.597	0.5304	0.5618
18	0.6908	0.6615	0.6758	0.7931	0.7188	0.7541	0.9649	0.8397	0.898	0.8942	0.75	0.8158	0.8733	0.8285	0.8503
19	***0.7782***	***0.7423***	***0.7598***	***0.8529***	***0.9062***	***0.8788***	***0.9248***	***0.9389***	***0.9318***	***0.9386***	***0.8629***	**0.8992**	***0.8988***	***0.8715***	***0.8849***
20	0.7899	0.3615	0.496	0.7941	0.8438	0.8182	0.9237	0.9237	0.9237	0.9184	0.7258	0.8108	0.8635	0.7827	0.8211
21	1	0.0154	0.0303	1	0	0	0	0	0	0.0714	0.0161	0.0263	0.597	0.5304	0.5618
22	0.6908	0.6615	0.6758	0.7931	0.7188	0.7541	0.9649	0.8397	0.898	0.8942	0.75	0.8158	0.8733	0.8285	0.8503
23	*0.7782*	*0.7423*	***0.7598***	*0.8529*	*0.9062*	***0.8788***	*0.9248*	*0.9389*	***0.9318***	*0.9386*	*0.8629*	**0.8992**	*0.8988*	*0.8715*	***0.8849***
24	0.7899	0.3615	0.496	0.7941	0.8438	0.8182	0.9237	0.9237	0.9237	0.9184	0.7258	0.8108	0.8635	0.7827	0.8211
25	0.7636	0.4846	0.5929	0.7667	0.7188	0.7419	0.8450	0.8321	0.8385	0.9100	0.7339	0.8125	0.8145	0.7686	0.7909
26	0.7216	0.5385	0.6167	0.7333	0.6875	0.7097	0.8537	0.8015	0.8268	0.9388	0.7419	0.8288	0.8150	0.7651	0.7892
27	0.749	0.7346	0.7417	0.7429	0.8125	**0.7761**	0.9558	0.8244	**0.8852**	0.9138	0.8548	**0.8833**	0.9043	0.8753	**0.8896**
28	0.7913	0.7731	**0.7821**	0.75	0.75	0.75	0.9722	0.8015	0.8787	0.8235	0.7903	0.8066	0.9062	0.8513	0.8779
29	0.6706	0.6577	0.6641	0.8235	0.8750	**0.8485**	0.9449	0.9160	**0.9302**	0.9386	0.8629	**0.8992**	0.8866	0.8593	**0.8727**
30	0.6752	0.6077	0.6397	0.7	0.6563	0.6774	0.8889	0.855	0.8716	0.9388	0.7419	0.8288	0.8105	0.7843	0.7972
31	0.7500	0.5538	0.6372	0.3636	0.7500	0.4898	0.8881	0.9084	0.8981	0.9320	0.7742	0.8458	0.8420	0.8061	0.8236
32	0.8532	0.8557	**0.8545**	0.6970	0.7188	0.7077	0.8603	0.8931	0.8764	0.8739	0.7823	0.8255	0.8293	0.8006	0.8147

Appendix-4: Precision, Recall, and F-measure Values on METUNER

run_id	PERSON			LOCATION			ORGANIZATION			TEMPORAL			Overall			ITER.
	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure	precision	recall	f1-measure	
1	62.13	66.59	64.28	64.80	70.88	67.70	43.84	61.94	51.34	54.49	25.26	34.52	60.49	60.46	60.47	239
2	57.95	63.55	60.62	58.86	63.99	61.32	64.09	57.53	60.63	69.90	42.28	52.69	60.23	59.32	59.77	176
3	75.15	74.22	74.68	73.78	68.58	71.08	76.70	61.72	68.40	73.88	69.72	71.74	74.92	70.63	72.71	382
4	69.25	57.93	63.09	80.49	55.46	65.67	56.10	50.36	53.08	77.10	35.69	48.79	71.38	53.98	61.47	100
5	79.11	77.78	78.44	80.20	77.02	78.58	77.57	69.69	73.42	80.32	66.92	73.01	79.03	74.00	76.43	259
6	73.92	79.20	76.47	66.75	72.65	69.58	65.82	71.90	68.73	68.20	59.60	63.61	69.30	73.77	71.47	156
7	83.28	82.61	82.94	72.68	75.25	73.94	72.03	66.51	69.16	80.66	76.34	78.44	77.00	74.95	75.96	315
8	85.09	78.65	81.74	73.16	69.51	71.29	69.52	68.70	69.11	80.74	66.70	73.06	77.13	72.47	74.72	177
9	76.47	75.35	75.90	76.15	71.38	73.69	66.43	62.51	64.41	65.80	67.75	66.76	72.73	70.34	71.52	308
10	71.06	78.08	74.41	64.69	72.45	68.35	55.52	65.36	60.04	44.49	64.90	52.79	62.40	72.02	66.86	169
AVG	73.34	73.40	73.26	71.16	69.72	70.12	64.76	63.62	63.83	69.56	57.52	61.54	70.46	68.19	69.14	