REPUBLIC OF TURKEY

AKDENIZ UNIVERSITY

# ANALYSIS OF NATURAL LANGUAGE PROCESSING TECHNIQUES AND DEVELOPMENT OF TURKISH NAMED ENTITY RECOGNITION TOOL FOR TRAVEL-TOURISM VOICE ASSISTANT

Deniz GÜL ÖZCAN

INSTITUTE OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

MASTER OF SCIENCE THESIS

DECEMBER 2020

ANTALYA

REPUBLIC OF TURKEY

AKDENIZ UNIVERSITY

# ANALYSIS OF NATURAL LANGUAGE PROCESSING TECHNIQUES AND DEVELOPMENT OF TURKISH NAMED ENTITY RECOGNITION TOOL FOR TRAVEL-TOURISM VOICE ASSISTANT

**Deniz GÜL ÖZCAN**

**INSTITUTE OF NATURAL AND APPLIED SCIENCES**

**DEPARTMENT OF COMPUTER ENGINEERING**

**MASTER OF SCIENCE THESIS**

**DECEMBER 2020**

**ANTALYA**

REPUBLIC OF TURKEY

AKDENIZ UNIVERSITY

INSTITUTE OF NATURAL AND APPLIED SCIENCES

# ANALYSIS OF NATURAL LANGUAGE PROCESSING TECHNIQUES AND DEVELOPMENT OF TURKISH NAMED ENTITY RECOGNITION TOOL FOR TRAVEL-TOURISM VOICE ASSISTANT

**Deniz GÜL ÖZCAN**

**DEPARTMENT OF COMPUTER ENGINEERING**

**MASTER OF SCIENCE THESIS**

This thesis unanimously accepted by the jury on 28/12/2020.

| | |
|---|---|
| Assoc. Prof. Dr. Ümit Deniz ULUŞAR (Advisor) | **e-imzalıdır** |
| Prof. Dr. Melih GÜNAY | **e-imzalıdır** |
| Assoc. Prof. Dr. Ahmet Cüneyd TANTUĞ | **e-imzalıdır** |

## ÖZET

**DOĞAL DİL İŞLEME TEKNİKLERİNİN İNCELENMESİ VE SEYAHAT-TURİZM SESLİ ASİSTANI İÇİN TÜRKÇE VARLIK İSMİ TANIMA ARACI GELİŞTİRİLMESİ**

**Deniz GÜL ÖZCAN**

**Yüksek Lisans Tezi, Bilgisayar Mühendisliği Anabilim Dalı**

**Danışman: Doç. Dr. Ümit Deniz ULUŞAR**

**Aralık 2020; 39 sayfa**

Bu tezde, seyahat sektöründe Türkçe dilinde bir seyahat asistanının doğal dil işleme teknikleri kullanılarak nasıl geliştirilebileceği konusunda çalışmalar yapılacaktır. Bu çalışmayı yaparken, literatürde daha önce yapılmış olan doğal dil işleme çalışmalarının araştırılması ve bu çalışmalarının Türkçe 'ye uyarlanabilirliğinin verilerle ölçülmesi hedeflenmiştir. Türkçe doğal dil işleme alanındaki akademik çalışmaların başarısının incelenmesi, bunların doğruluk oranlarının nasıl arttırılabileceğinin saptanması ve bulunan sonuçların pratik uygulamalara nasıl uygulanabileceğinin çalışması yapılacaktır. NLP tekniklerine odaklanan akademik çalışmalara odaklanılarak ve bir Türk Mobil Seyahat Asistanında bu teknikler kullanılarak elde edilen sonuçları karşılaştırılacaktır.

Doğal dil işleme, doğal dilde yazılmış metinlerin veya konuşulmuş seslerin işlenerek bilgisayar ile anlamlandırılması üzerine çalışmalar yürütülen, bilgisayar bilimi ve dil bilimciliğinin bir alt bilim dalıdır. Türkçe doğal dil işleme ise bu alanda yapılan çalışmaların Türk dil ve morfolojik yapısına göre uyarlanması ve gerektiğinde daha önce kullanılmamış tekniklerin kullanılarak dilin yapısına uygun olarak yeni yöntemlerin keşfedilmesidir.

Bilgisayarların dili "anlama" terimi, insanın anladığı gibi değildir. Bilgisayarlara, dili anlamak için verilen modelleri kullanmak üzere denetimli, yarı denetimli ve denetimsiz yaklaşımlar kullanan modeller ve veriler sağlanmalıdır. Burada başarı anlama oranları, bir bilgisayarın dili konuşan bir insana kıyasla niyeti ne kadar iyi anladığı olarak tanımlanabilir. Bu tanımı temel alarak, NLP' deki en son modelleri kullanarak, dili anlayan bilgisayarların başarı oranları, mevcut bilgisayarların ve verilerin işlem gücünün artmasıyla arttı.

Literatür tarandığında Türkçe doğal dil işleme alanında birçok çalışma yapıldığını görmekteyiz. Ses dalgalarının incelenerek ses tanıma ve yazıya aktarılması çalışmalarına konuşmadan yazıya yazıdan konuşmaya işlemleri denilmektedir. Bunun yansıra biçimbilimsel, sözdizimsel, anlamsal çözümleme gibi yöntemler kullanılmakta ve geliştirilmektedir. Bu alanda yapılan diğer çalışmalar otomatik yazım hatalarının denetlenmesi ve düzeltilmesi, dil çeviri gibi pratik, her daim elimizin altındaki alanlara da yaygınlaşmıştır. Büyük verinin artmasıyla birlikte bilginin ulaşılabilirliğinin zorlaşması ile bilgi çıkarımı, bilgi getirimi, soru cevap sistemlerinin geliştirilmesi, özet çıkartma gibi uygulamalar doğal dil işleme tanımı altında gerekli yerini bulmuştur.

Bu alanda daha önce de sesli asistan geliştirme çalışmaları yapılmıştır. Ancak yaygın olarak kullanılmamıştır ve araştırma aşamasında kalmıştır. Doğal dil işleme alanında çalışmaların yürütülebilmesi için güncel olarak makine öğrenimi yöntemleri kullanılmıştır. Makine öğrenmesi tekniklerinin bu alanda kullanılabilmesi için de mevzu bahis alanda büyük veriye ihtiyaç duyulmaktadır. Konu özelleştikçe veriye ulaşmak daha da zorlaşmıştır. Bu tez kapsamında da seyahat alanında sesli niyetin toplanarak ve üzerinde çeşitli DDİ teknikleri kullanılarak veri haline getirilmesi ve bu verinin makine öğrenmesi yöntemlerine girdi olarak sağlanabilmesi hedeflenmiştir. Sesli asistan alanında çalışmalar yapılmasına rağmen seyahat alanında bugüne kadar çalışma yapılmamıştır. Bu da taranan literatürden gözlemlenmektedir.

Bu çalışmada ses yoluyla alınan niyetin içerisinden gürültü (noise) olarak bildiğimiz model üzerinde etkisi olmayan kısımlar arındırılarak varlık ismi tespitine gönderilecektir. Varlık tespiti aşamasında bu cümlenin içerisinden anlamlı veriler alınıp seyahat edilecek yer, yön, tarih, kişi vb. dataları çıkarıp bunları birbirleriyle anlamlarını inceleyip modele oturtacaktır. Bu aşamada kişi isteğine göre listelenen sonuçlardan hangisini seçtiğini de ses yoluyla belirtecektir ve rezervasyon sayfasına ya da kişinin bilgilerinin istendiği sayfaya yönlendirilecektir. Buradan sonra kişinin bilgileri sesli soru cevap sistemi ile alınıp sesten yazıya çevrilecektir. Makine öğrenmesi algoritmaları çağrı merkezi çalışanlarının ve müşterilerinin gerçek konuşmalarından da öğrenerek veri kümelerini geliştirecektir ve bir süre sonra kendi kendine birçok sesli niyeti anlayıp çözüme ulaştıracak yapıya gelecektir.

Türkiye’de doğal dil işleme alanında hali hazırda var olan kütüphanelerin (Zemberek) ve web servislerinin (İTÜ NLP Web Service) sağladığı yöntemler araştırılacaktır. Aynı zamanda son yıllarda Transfer Öğrenimi (transfer learning) yöntemleri de doğal dili anlama ve işleme alanlarında hızlı ilerlemeler kaydetmiştir. Bunlar ile deneylerle kullanılabilirliği araştırılacaktır. Bu yöntemlere ek geliştirmeler yaparak çalışmanın tamamlanması öngörülmüştür. Veri analizi yöntemleri ile daha önceden alınmış biletlerin konuşmalarından benzerlik algoritmaları ile şimdiki konuşmanın anlamını çıkarmaya çalışmak da bir gidiş yolu olarak öngörülmüştür. Bunlar AR-GE sonucunda başarı oranına göre karar verilerek en doğru kararı veren algoritma ile devam edilecektir.

**ANAHTAR KELİMELER:** Makine Öğrenmesi, Türkçe Doğal Dil İşleme, Ses Tanıma, Seyahat ve Turizm Alanında Doğal Dil İşleme, Türkçe Biçimbilimsel Analizi, Türkçe’de Varlık İsmi Tanıma,

## ABSTRACT

**ANALYSIS OF NATURAL LANGUAGE PROCESSING TECHNIQUES AND DEVELOPMENT OF TURKISH NAMED ENTITY RECOGNITION TOOL FOR TRAVEL-TOURISM VOICE ASSISTANT**

**Deniz GÜL ÖZCAN**

**MSc Thesis in Computer Engineering**

**Supervisor: Assoc. Prof. Dr. Ümit Deniz ULUŞAR**

**December 2020; 39 pages**

The aim of this thesis is to conduct studies on development of a Turkish language mobile travel assistant in the travel industry using natural language processing techniques. While conducting this research, it is aimed to investigate the natural language processing done so far in the literature in languages other than Turkish and to measure the adaptability of these studies to Turkish. Turkish natural language processing studies provide resources for both our own language and other Turkish languages. Examining the success of academic studies in the field of Turkish natural language processing, determining how their accuracy rates can be increased and how the results found can be applied to practical applications will be done.

Natural Language Processing or NLP is defined as the system of processing human language and understand the intention. It is a sub-field of artificial intelligence and focuses on topics such as speech recognition, morphological processing, disambiguation, dependency parsing and named entity recognition of the language. Although being researched under NLP, these topics are comprehensive and comprises of rich research all by themselves.

The term of computers ‘understanding’ the language is not like the way human understand it. The computers should be provided with models using supervised, semi supervised and unsupervised approaches and data to use those given models to understand the language. Here, the success rates of understanding can be defined as how close a computer understands the intent comparing to a human speaking the language. Using the state-of-the-art models on NLP, the success rates of computers understanding the language increased with the increase of processing power of available computers and data.

NLP can be used on areas such as sentiment analysis, machine translation, automatic speech recognition systems on various domains. In this article, I will present the academic studies focusing on the state-of-the-art NLP techniques and compare the results achieved by using those techniques on a Turkish Mobile Travel Assistant.

When the literature is reviewed, we see that there are many studies in the field of Turkish NLP. In this context, methods such as recognizing the waves of the sound and transferring them to the text in the computer, text vocalization, morphological analysis-production, syntax analysis, semantic analysis are used and developed. Applications such as checking / correcting spelling errors, machine translation, information extraction, information retrieval, development of question-and-answer systems, summary extraction are collected under the definition of natural language processing.

Voice assistant studies have been conducted before in Turkish. However, it was not widely used and remained in the research phase. In order to carry out studies in the field of NLP, machine learning methods are commonly used. In order to use machine learning techniques in this area, big data is needed in the subject. The data is not available due to the fact that the studies have remained at the research level or the data has been privatized by privately held companies. As the subject become more specific like travel sector, it becomes more difficult to access the data. Within the scope of this thesis, it is aimed to collect the voice intention in the field of travel and convert the voice into data using various NLP techniques and to provide this data as an input to machine learning methods. Although there are studies in the area of voice assistant, no studies have been carried out in the field of travel until today.

Classification methods and machine learning models will be used during the development of an NLP Travel assistant. In this work, The Named Entity Recognizer (NER) tool utilizes transfer learning methods, namely Google BERT and ELECTRA. These models will be fine-tuned with 4 types of data. Those fine-tuned models will be evaluated with a unique test data that is common to all those data sets. The extended travel NER tool will be able to tag PERSON, LOCATION, ORGANIZATION, DATE, TIME entities.

**KEYWORDS:** Machine Learning, Turkish Speech Recognition, Turkish Natural Language Processing, Turkish Natural Language Understanding, Turkish Morphological Analysis, Turkish Named Entity Recognition

## ACKNOWLEDGEMENTS

Türk milletinin dili Türkçedir. Türk dili dünyada en güzel, en zengin ve en kolay olabilecek bir dildir. Onun için her Türk, dilini çok sever ve onu yüceltmek için çalışır… (1929)

Mustafa Kemal Atatürk

I am enthusiastically presenting this work to the world of science, in order to contribute to the willing researchers who are working in this field as a result of my respect for the Turkish language and my indispensable passion for Turkish success. Although the studies in Turkish are gaining momentum in the latest years, more attention is needed from the science world. I wish this work will be an inspiration to attract the attention to the natural language processing in Turkish.

I would like to thank to Assoc. Dr. To Gülşen Cebiroğlu Eryiğit and Assoc. Prof. Dr. Ahmet Cüneyd TANTUĞ whose works shed light on me with studies in the literature and to my advisor, Assoc. Prof. Dr Ümit Deniz ULUŞAR who gave me all kinds of support during my master's education and my thesis work and who is a role model for my future academic career.

I send my endless love and gratitude to my mother, father, husband and my daughter Derin Elis, who supported me with my thesis discussion, in the hope of compensating for the times I was away from them. I hope this thesis work and publications contribute to scientific research.

## LIST OF CONTENTS

## TEXT OF OATH

I declare that this study "Analysis of Natural Language Processing Techniques and Development of Turkish Named Entity Recognition Tool for Travel-Tourism Voice Assistant", which I present as master thesis, is in accordance with the academic rules and ethical conduct. I also declare that I cited and referenced all material and results that are not original to this work.

28/12/2020

Deniz GÜL ÖZCAN

[illegible]

# ABBREVIATIONS

**<u>Abbreviations</u>**

| | |
|---|---|
| NLP | : Natural Language Processing |
| API | : Application Programming Interface |
| SVM | : Support Vector Machines |
| CRF | : Conditional Random Fields |
| NER | : Named Entity Recognition |
| POS | : Part of Speech |
| SFST | : Stuttgart finite state transducer tools |
| METU | : Middle East Technical University |
| BLSTM | : Bidirectional Long Short-Term Memory |
| CoNLL | : Conference on Natural Language Learning. |
| MUC | : Message Understanding Conference |
| GLUE | : General Language Understanding Evaluation. |
| BERT | : Bidirectional Encoder Representations from Transformers |
| NLU | : Natural Language Understanding |
| MLM | : Masked Language Modeling |
| IMST | : ITUMetuSabancıTreebank |

## LIST OF FIGURES

## LIST OF TABLES

# 1. INTRODUCTION

Natural Language Processing or NLP is defined as the system of processing human language and understand the intention. It is a sub-field of artificial intelligence and focuses on topics such as speech recognition, morphological processing, disambiguation, dependency parsing and named entity recognition of the language. Although being researched under NLP, these topics are comprehensive and comprises of rich research all by themselves.

The term of computers 'understanding' the language is not like the way human understand it. The computers should be provided with models using supervised, semi supervised and unsupervised approaches and data to use those given models to understand the language. Here, the success rates of understanding can be defined as how close a computer understand the intent comparing to a human speaking the language. Taking this definition as a base, using the state-of-the-art models on NLP, the success rates of computers understanding the language increased with the increase of processing power of available computers and data.

It is observed that mobile applications, which are able to respond quickly to user requests in every field, have increased since the use of mobile applications very widely today. In today's mobile world, we are able to issue commands both by typing and by speaking, in order to receive service from the phone. Applications of personal assistants developed by global companies are used by many of people.

We observe that personal assistants make considerable progress in many general subjects using natural language processing methods. Although the travel industry is developing rapidly and keeping up with the mobile age with the applications made to date, it is observed in the literature that the research and applications in Turkish language are insufficient.

NLP can be used on areas such as sentiment analysis, machine translation, automatic speech recognition systems on various domains. In this article, I will present the academic studies focusing on the state-of-the-art NLP techniques and compare the results achieved by using those techniques on a Turkish Mobile Travel Assistant.

A mobile travel assistant naturally consists of a pipeline that includes speech recognition such as speech to text and text to speech parts as well. The pipeline consists of real time modeling and increasing the model performance of the model while retrieving new data from the user. The pipeline consists of a classifier which simply classifies the given intent into some predefined intentions. This part uses supervised learning methods. This whole pipeline is implemented throughout this work, but main focus was on NER tool for travel domain.

Turkish natural language processing studies provide resources for both our own language and other Turkish languages. Examining the success of academic studies in the field of Turkish natural language processing, determining how their accuracy rates can be increased and how the results found can be applied to practical applications will be done.

When we review the literature, we see studies on various aspects of Turkish natural language processing. In this context, methods such as recognizing the waves of the sound and transferring them to the text in the computer, text vocalization, morphological analysis-production, syntax analysis, semantic analysis are used and developed. Applications such as checking / correcting spelling errors, machine translation, information extraction, information retrieval, development of question-and-answer systems, summary extraction are collected under the definition of natural language processing. Voice assistant studies have been conducted before in Turkish. However, it was not widely used and remained in the research phase. In order to carry out studies in the field of NLP, machine learning methods are commonly used. In order to use machine learning techniques in this area, big data is needed in the subject. The data is not available due to the fact that the studies have remained at the research level or the data has been privatized by privately held companies. As the subject become more specific like travel sector, it becomes more difficult to access the data. Within the scope of this thesis, it is aimed to collect the voice intention in the field of travel and convert the voice into data using various NLP techniques and to provide this data as an input to machine learning methods. Although there are studies in the area of voice assistant, no studies have been carried out in the field of travel until today.

The Turkish intention will be acquired using speech recognition algorithms. After transforming the speech to text, it is necessary to gather only the words needed with the help of normalization, vocabulary and morphological analysis. It is envisaged that the resulting words will be compared with datasets and placed on the domain model needed by online travel companies in the industry. Thus, voice intent will become an API format that web applications can communicate with.

At every stage within the scope, achievement will be increased by machine learning methods. There are many stages of natural language processing called NLP pipeline. One of the most helpful stages in natural language processing is named entity recognition.

The aim of this thesis is to develop a token classification tool for a Turkish mobile travel assistant specifically focusing on tourism/travel domain. Token classification is studied under the name NER. Named-entity recognition (NER) can be defined as the process of identifying and categorizing the named-entities such as person, location, product and organization names, or date/time and money/percentage expressions in unstructured text. Travel NER tool will be the one that specifically targets the tourism domain. While conducting this study, it is aimed to investigate the NLP research done so far in the literature in languages other than Turkish and to evaluate the applicability of these studies to Turkish. The results of NER tool will be compared with results of state-of-the-art research in token classification area.

## 2. LITERATURE REVIEW

There are some works completed or at least researched during the development of a mobile personal assistant. These personal assistants were using the learning methodologies of that time of writing. One of those works is the article named “A Mobile Assistant for Turkish”. In their article Çelikkaya et. al. explains the classification methods and machine learning models they used during the development of an NLP assistant. They state that normalization process is necessary for the purpose of capitalization of Proper Nouns and Accent Normalization. They use Conditional Random Fields (CRFs) in their work. This model is a statistical modeling method. They extended their data set with currency and time gazetteers. They added 750 tagged queries and commands. The following features are considered for training the NER tool: whole word, word stem, POS tag or word, noun state, proper noun, inflectional groups, lower-upper case, sentence begin, and existence in gazetteers. They extended the NER tool adding the capabilities of tagging PERSON, LOCATION, ORGANIZATION, TIME, MONEY, and PERCENTAGE entities. Bag-of-words (bag-of-n-grams) model is used to represent the data as numerical values. More on this model can be found in the article but it is widely considered in document classification methods. It grabs attention specifically while training a classifier if frequency of occurrence of each n-gram is used as a feature. The performance of the classification methods such as K- Nearest-Neighbor, Decision Tree, Logistic Regression, Multinominal NaiveBayes, Support Vector Machine, Bayesian Network are evaluated. 10-fold cross validation is used to compare the performance of the classification methods because of small scale of data. They state that 1-gram model produces better results compared to 2-gram and 3-gram models.(Çelikkaya et al. 2014)

After 3 years, another work had been completed by the same authors for a mobile personal assistant using state of the art methodologies. Celikkaya et. al. in their article on “Use of NLP Techniques for an Enhanced Mobile Personal Assistant: The Case of Turkish” introduce a mobile assistant. This assistant is developed to classify tasks, such as sending SMS messages and making calls. With location and email services, user can get directions and send emails. Their application included various stages like NLP, query classification and parameter extraction. This application does not only inform the user but also performs the intended task. Different types of datasets are annotated for classification and parameter extraction, domain adaptation of named entity recognition and syntactic parsing NLP modules. In this study, the application was developed by using Google's speech recognition API for Android. MaltParser is a system, developed by Johan Hall, Jens Nilsson and Joakim Nivre, for data-driven dependency parsing, which can be used to induce a parsing model from treebank data and to parse new data using an induced model. In their work, Maltparser was used as syntactic parser. IMST dataset is used for machine learning. For the statistical classification experiments, training data was used with 10-fold cross-validation. Weka tool is a java-based machine learning tool for algorithms like Logistic Regression, Support Vector Machines (SVM), Naive Bayes, Decision Tree. Those algorithms were experimented throughout this study. Preprocessing and SVM yield the best results. They also measured the impact of NLP to the query classification. Those impact results are between 0.2 to 10.7 percentage points which highly depends on the algorithm. NLP is also crucial for the parameter extraction stage. This multistage approach performs 70.8% accuracy score although the annotated data is limited.(Eryigit 2017)

There are other works developed for Turkic languages. These works and their source codes are available online for public use. These works mostly concentrates on morphological structure of Turkish and Turkic languages. Akın et. al. in their article named "Zemberek, an open-source NLP framework for Turkic Languages" explains Zemberek project in detail. First of all, it is developed in Java language. Also, it is open source and because of development language being java, it is platform independent. Developer can utilize the NLP framework for Turkish and all Turkic languages. Zemberek library is consisted of two main parts as language structure information and NLP operations. Framework helps developer with common NLP tasks such as morphological parsing, spell checking and stemming. Developer can also use Zemberek in generative tasks other than morphological ones such as word construction, word suggestion. There are more tasks in the framework which is beyond this thesis. Zemberek is available on GitHub. Library stores the alphabetical information like letters, vowels etc. in a simple text-based file. Suffix groups and individual suffix information are stored in a configuration file. Morphological parser of Zemberek works on the simple principle of suffixes followed by certain suffixes. It finds the possible root and suffixes of a given word. We can define its structure as a dictionary based top-down parser. Zemberek utilizes root dictionary-based parser, so it requires a root finding mechanism. A Root word tree is used as a dictionary and it allows different kinds of root selectors to be implemented easily. Zemberek Library is also used in Turkish spell checker algorithms.(Af et al.)

Another morphological study is completed by Çöltekin namely "A Freely Available Morphological Analyzer for Turkish". In this work, he presents a two-level finite state Turkish m. analyzer named TRMorph. It is a free, complete and accurate Turkish morphological analyzer. Analyzers implemented as of those time this work completed, are generally rule-based systems which are typically implemented using finite state transducers (FSTs). This analyzer is implemented as two-level using Stuttgart finite state transducer tools (SFST). The author also included a large lexicon adapted from Zemberek. SFST uses a language based on regular expressions. TRMorph is compatible with UTF-8 encoding. TRMorph currently only analyzes words and its strength lies in its availability. Free tools and resources are used in this analyzer implementation, so it is free to use modify and distribute. It forms a basis for implementing different tasks and other morphological analyzers for other Turkic languages.(Çöltekin 2010)

Çöltekin continues his works on morphological structure and contributes to NLP community with his works explained in this article named "A set of open-source tools for Turkish natural language processing". In this article, he introduces new tools for morphological tasks in Turkish like stemming and lemmatization, guessing unknown words etc. These tools are presented as an update on major changes on TRMorph as we summarized above. The article starts with Turkish is being a morphologically complex language. A language is defined as complex when the conventional rules of stems and suffixes does not hold many times. In addition, a word can carry way more different meanings comparing to the whole word. Another difficulty while analyzing a language morphologically is the issue of data scarcity that affects the machine learning methods used during the trainings. One solution to these kinds of problems offered so far is to divide the word into inflectional groups and analyze the morphology. The article continues with the updates on TRMorph. Morphological tag set is completely revised in this version. It was using a modified version of lexicon taken from Zemberek. A new lexicon is introduced. Earlier version of TRMorph was implemented in SFST. The new version has been updated with languages from Xerox called lexc and xfst. He introduces a new morphological disambiguation model and experiments these 3 studies. A rote-learning disambiguator which performs worst. Making use of IGs (Inflectional Groups) cannot perform as the last one. making use of analysis without root and model performs the best of all three. These updated and additional tools are mostly demanded by users which eventually makes TRMorph a more usable system for Turkish NLP community. Active development still continues on TRMorph to bring up new features and functionality.(Çöltekin 2014)

So far, we have analyzed literature on morphological parsing of Turkish, yet this is another morphological work focuses on dependency parsing of Turkish. Eryiğit et. al. in their article on "Dependency Parsing of Turkish". Almost every researcher state that Turkish is a free constituent order language with a rich agglutinative morphology. Turkish is predominantly head-final language. In this work, they investigate how different design choices affect data-driven parsers. They use data from the recently released Turkish Treebank, they investigate the impact of different design choices in developing data-driven parsers. The issues addressed in the experiments can be summarized as treatment of morphology in syntactic parsing, importance of lexicalization and parsing methodology. So far, Turkish has been represented by splitting the words into inflectional groups called IG's. The root and the derivational parts are represented with different IG's. These IG's are separated by derivational boundaries called DB's. They stated that parsing accuracy can be improved dividing the words into smaller units than words. Their proposed system contains two naive parsers, called baseline parsers and data-driven parsers. Data-driven parsers are consisted of a probabilistic parser using conditional probabilistic model and a classifier-based parser using discriminative learning. They evaluate their system with ASU: unlabeled attachment score and ASL: labeled attachment score. The metric is the proportion of IG's to ASL's. They evaluate their parsers by implementing another three parsers. First two works as attaching the words to parser, the third one works with a rule-based system. Third one uses a stack to store partially processed tokens. They experimented different approaches which are detailed in their article, however they concluded that the statistical approach is a well-studied one in all. In statistical approach, the tagged sentence is the input, the probability dependency tree is the output. They concluded that the IG-based model is the most successful one in all parsers and word-based models introduced so far. IG-based model not only improves the ASU accuracy but also word-to-word accuracy. They could achieve the highest accuracy by combining morphological information, and lexicalization with IG-based representations for parsing the Turkish Treebank (Eryiğit et al. 2008).

There are also some works related to creating Turkish corpus in the literature. The following two article summarizes these works mostly completed by the same author in different years. Sezer et.al in their article written originally in Turkish as "TS Corpus: Herkes için Türkçe derlem" presents the work of TSCorpus in detail which is a large Turkish corpus consisting of 491M tokens. The aim of the work is to preprocess the available unprocessed corpus available on Turkish and tag the token with their POS-Tags, morphological tagging and lemmas. TSCorpus is the first corpus which is available and can be queried on a user-friendly web interface. It is created with the CWB/CQP structure.

BOUN Web Corpus data which was available online at the time of the work, is used as data source. An averaged perceptron-based morphological disambiguator for Turkish text is used for morphological tagging and disambiguation. The result of the analyzer is converted into CWB/CQP (IMS Open Corpus Workbench). Open Corpus Workbench is a framework for querying and managing large corpora. Large means words ranging from 10 million to 2 billion words with their linguistic annotations. In its center lays an efficient and flexible query processor called CQP. One can use TSCorpus for simple queries and using CQP queries. This is achieved by the framework of CWB provides (Sezer et al. 2013).

We have discussed the works on Turkish morphology and dependency parsing so far and almost all of those works have been completed with corpora created by other researchers. In this article, we will look into how these corpora are created. Corpora means the main body or a mass of a structure in anatomy. In literature it means a collection of written texts or body of writings. Sezer in his article on “TS Corpus Project: An online Turkish Dictionary and TS DIY Corpus” presents an online dictionary named TS Corpus. In this corpus he concentrates how input is hyphenated, trigrams applied and collocated. TSCorpus is a free tool that a researcher can use to build a corpora or linguistic datasets by own. It includes a do-it-yourself corpus software with which one can create a corpus, add or edit content, run queries. The most important feature is the presentations of the collocations for the query key. Spell book module that uses Zemberek as spell-checker is another future coming with this version. Sezer solves the problem of creating one to prepare their own corpus with new feature TS DIY Corpus. It is online available, easy to use, flexible and includes auto processing tools. Python Django is used for the main framework. The software includes some important scripts called TS Tokenizer and POS tagger. Users can save their queries and results in many formats and export those results (Sezer 2017).

After analyzing the morphological structure and how to create a Turkish Corpus, we need to dive in to the NER problem. For this reason, there is a work in the literature which is widely read and referenced because it successfully defines and summarizes the terms and works done so far for NER and classification problem. Nadeau and Sekine in their article on “A survey of named entity recognition and classification” present a survey of research made in the named entity recognition and classification area of natural language processing. They approach to the study of NER by analyzing the language factor, textual genres or domain factor and entity type factors. They provide the used learning methods on research mainly being supervised, semi-supervised and unsupervised learning. They define these learning methods in terms of the usage in NERC. Semi-supervised learning is performed by reading a large, annotated corpus. Based on the discriminative features it creates disambiguation rules. Most of NER problems were solved or at least experienced with those SL models like Decision Trees, SVM and Conditional Random Fields (CRF), etc. Bootstrapping method is a resampling technique by simply sampling a dataset with replacement. “bootstrapping” method is the main technique used in semi-supervised learning. A set of seeds are provided to the system to start the learning process. Clustering is automatically discovering natural grouping in data. Clustering is typical approach in unsupervised learning. The methods are basically based on lexical tools (e.g. WordNet), lexical patterns and statistics calculated on a large unannotated corpus. The authors continue with the feature space for NERC systems. Each feature, or column, represents a measurable piece of data that can be used for analysis. It represents characteristic attributes of words. Generally, the NERC pipeline is an application of the rule system over the features. word-level features, list-lookup features, document and corpus features are some of them. Word-level features describe word case, punctuation, character etc. List-lookup features are generally called “gazetteer”, “lexicon” and “dictionary”. Large collections of documents are extensive sources of features. All these features are analyzed in deep in the article with the usage examples. The article continues with the evaluation methods of NERC. They explained three main scoring techniques which are used in MUC, IREX, CONLL and ACE conferences (Nadeau et al. 2007).

There are rule based NER systems developed by Turkish authors in 2009’s. Küçük et.al in their article on “Named Entity Recognition Experiments on Turkish Texts” presents a rule-based Named Entity Recognition system for Turkish. They define NER task as an information extraction task. NER is defined as the extraction of certain attributes from the sentence. NER task can be time consuming with supervised learning systems because of the necessity of training corpora. Therefore, bootstrapping algorithms which is a semi-supervised learning type can be used to bypass those preprocessing tasks. This algorithm is basically feeding the system with a set of examples or patterns, the system iteratively induces new patterns using those seed set. The morphological analyzer implemented by Küçük et.al considers only the noun inflections. The objects that conform to the pattern are extracted from the texts. They built the system for Turkish news texts and evaluated on news articles gathered from METU Turkish corpus. They created a dictionary of person names, locations and organizations. They also prepared patterns for the named entities including dates and times. They evaluate the results comparing the results with state-of-the-art research on NER and the best performance is 91.56% f-score and overall system scores 78.7% f-score in average. These scores can be improved by adding algorithms like automatic rule induction (Küçük et al. 2009).

In 2012, rather than using rule-based system, NER task is experimented using Conditional Random Fields (CRF). Şeker et.al in their article on “Initial explorations on using CRFs for Turkish Named Entity Recognition” presents a Turkish NER (named entity recognition) model using CRF and trained with morphological and lexical features. They start with the morphological complexity and structure of Turkish and how they applied their preprocessing work on this agglutinative language. In Turkish, the position of the word does not tell us enough information if that word is a named entity or not because of its being a free word order language. That is why, they first tokenized their data to represent each word as a token. They use a two-level morphological analyzer to analyze each word. They input the analyzer results to the morphological disambiguator to get the analysis with highest probability. They prepared two gazetteers named base and generator gazetteers. Then, they prepared the feature vectors using the raw data, gazetteers list and morphological processing. They used CRF++ that is a free and open-source tool for CRF implementation. The results are evaluated using both CoNLL and MUC. They compared their results with previous works on NER. As a result, they state that morphological features can raise the performance. They also compared the results and the evaluation metrics of recent NER work in Turkish. Person and location names gazetteers are available to the researchers. These state-of-the-art results forms a baseline for Turkish NER research (Şeker et al. 2012) .

Yeniterzi et.al in their article on “Turkish Named Entity Recognition” first summarizes the previous works on Turkish NER. She starts with the representation of data used by the researchers. Mostly used representation types are raw named entity tags, BIO short for inside outside beginning and BILOU adds a last token with L and a unit length. She continues with the evaluation of NER performance. So far, the researchers evaluated their NER results using three types of metrics. These metrics such as MUC, CoNLL, and ACE are named after the NER conferences that they are first used. ACE is not commonly used. MUC and CoNLL metric both uses the f-measure to report the finalized score. The article continues with types of datasets being formal texts like news, articles and books, informal texts like social media text, tweets. There are some preprocessing techniques applied before entity recognition task like normalization of sentences, tokenization and morphological analysis. NER systems usually use word-level tokenizers. Morphological analysis is used commonly as a preprocessing step. Data sparsity has been a challenging issue for NER task. To deal with this problem, stems or root words are used by some NER systems. If researcher is dealing with the text in informal domains, normalization is a preprocessing technique to format text into more formal way. The article continues with the approaches applied to NER task. These approaches are analyzed in three as hand-crafted rule-based systems, machine learning based systems, and hybrid systems. Hybrid systems are nothing but the combination of the first two systems. Conditional Random Fields (CRF), word embeddings and Bayesian learning approach have been experimented in some NER tools. From these approaches, machine learning based approaches achieved the highest performance among all (2018).

Okur et.al in their article on “Named Entity Recognition on Twitter for Turkish using Semi-supervised Learning with Word Embeddings” presents a neural network based semi-supervised approach using word embeddings for the NER task on Turkish tweets. They used both unsupervised learning using unannotated large corpus and supervised learning with annotated Turkish twitter data. Their aim is to learn distributed word representations, or word embeddings, in continuous vector space where semantically similar words are expected to be close to each other. Word vectors trained on large un- labeled Turkish corpus can provide additional knowledge base for NER systems trained with limited amount of labeled data in the supervised stage. At first stage, they used the continuous Skip-gram model to obtain semantic representations of Turkish words. At the second stage, we exploited these word embeddings together with language independent features in order to train our neural network on labeled data. Their proposed system outperforms the state-of-the-art results on both Turkish Twitter data sets, even without using gazetteers. They achieved their best performance results with Turkish word embeddings obtained from their Web-Tweets corpus, when they apply normalization on tweets and keep the capitalization as a feature (Okur et al. 2016).

Recently, machine learning algorithms have been applied to Turkish NER and Tantuğ’s work is one of those studies which applies BLSTM on NER problem. In his article on “Turkish Named Entity Recognition with Deep Learning” for the case of Turkish NER, uses Bidirectional Long Short-Term Memory (BLSTM) method and compares it previous work. As for the previous works, NER is known as the study of various subject related entities to be recognized from the text. In a work HMM (Hidden Marcov Models) is used for NER of Turkish on a tagged data. In another work, the most used verbs are collected, and a rule-based system is produced resulting a %78.7 score of success on updated texts. Using the morphological structure of Turkish language an unusual method such as Conditional Random Fields-(CRF) is used by an author referenced and reached to a %88.9 score. The highest score achieved using CRF method is %92.15 in which the lexical and morphological properties of Turkish language is both used. In one work, a closely related study is conducted using DBLSTM methods and reached to a %91.30 F1 score. Word Vectors/Word Embeddings and Character Embedding and morphological embedding is used in this BLSTM study and reached to the highest score of %93.59 F1 (Gunes et al. 2018).

Available data grows exponentially, and those data cannot be handled manually. Using publicly available data Sahin et.al in their article on “Automatically Annotated Turkish Corpus for Named Entity Recognition and Text Categorization using Large-Scale Gazetteers” presents a dataset called TWNERTC. This dataset differs from others because the tokens are annotated automatically. Turkish Wikipedia dumps are used as text source. Freebase was a large collaborative knowledge base consisting of data composed mainly by its community members. It was an online collection of structured data harvested from many sources. It is depreciated as of now. Şahin et.al used freebase to construct a large-scale gazetteer. Freebase provides the domain information. Domain-independent and domain-dependent methodologies are utilized to solve noisy and ambiguous data. They made all versions of datasets publicly available through their project webpage. They contribute to the NER community with their Turkish corpus publication. Six different types of corpus are created using various reduction methodologies and entity types. They analyzed the corpus and compared their corpus against human annotators. Throughout the article, they explain the process and methodologies to construct the datasets and to eliminate noisy and incorrect data. They also provide statistics about dataset content. They concluded this work that they could achieve the success scores with their automatically annotated corpus that is very close to human annotated corpus (Sahin et al. 2017) .

Recent works showed that transfer learning has proven state of the art success. The foundation of success lays back to the works completed way before in 2010 analyzed under the article named “A Survey on Transfer Learning”. Pan et.al in their article centers around sorting and assessing the current advancement on transfer learning classification, regression, and clustering problems. In this study, they examine the connection between transfer learning and other related AI strategies, for example, domain adaptation, multitask learning and sample selection bias. Likewise, they investigate some possible future issues in transfer learning research. The greater part of traditional machine learning techniques expect that the distributions of the labeled and unlabeled information are the equivalent. Transfer learning, conversely, permits the areas, undertakings, and distributions utilized in preparing and testing to be different. They state that traditional machine learning techniques aim to learn any task from scratch, while where the latter has less high-quality training data, transfer learning techniques aim to transfer information from certain previous tasks to a target task. The authors give the definitions and categorization of transfer learning methodology in terms of mathematical formulas. They categorize transfer learning with three questions as what to transfer, how to transfer and when to transfer. The answers to these questions define the type of transfer learning and how to apply it to the data. They also warn about the negative transfer. When the source and target domain is not related, brute force knowledge transfers may reduce the performance of learning in the target task. When the learning in target domain is affected even negatively, then this is mostly called as negative transfer. They divide transfer learning into three named Transductive Transfer Learning, Inductive Transfer Learning and Unsupervised Transfer Learning. They also compare some cases using the transfer learning and other machine learning methodologies (Pan et al. 2010) .

Now that we have learned the transfer learning and analyzed types of transfer learning, we can move to the recent developments. Being introduced in 2018, BERT grabs attention to experiment some NLP tasks like token classification. Devlin et.al in their article on “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding” explains the BERT which Is standing for Bidirectional Encoder Representations from Transformers. This is a new language representation model that differs from the other pretrained language representation models with it being bidirectional pre-training. It is designed to pretrain bidirectional representations from unlabeled text. It performs deep traverse through the whole text on both left and right context that ensures its being bidirectional in all layers. Adding only one output layer is mostly enough to perform state of the art results on various tasks. They also emphasize BERT to be simple besides empirically powerful. There are the two existing strategies for applying pre-trained language representations: feature-based and fine-tuning. BERT uses the latter one. BERT uses masked language modeling technique to enable pretrained deep bidirectional representations. One can achieve the state-of-the-art performance on sentence level and token level tasks by just fine tuning. BERT advances the state of the art for eleven NLP tasks such as GLUE, General Language Understanding Evaluation. They continue the article with detailed implementation of BERT. There are two steps in their framework: Pre-training and fine-tuning. The distinction comes with BERT is its being unified architecture across different tasks. BERT’s model architecture is a multi-layer bidirectional Transformer encoder (Devlin et al. 2019).

We will continue with another popular model created by the company OPENAI. Radford et.al in their article on “Improving Language Understanding by Generative Pre-Training” explore a generative-pretraining model for language understanding tasks. Unsupervised pre-training and supervised fine-tuning are combined to form a semi-supervised approach. Their main objective is to learn a universal representation that is transferrable to common tasks with a little adaptation. They use the Transformers as the model architecture which is proven to be well-performing on various tasks such as, document generation, machine translation and syntactic parsing. Task specific input adaptations are used during transfer. These adaptations are derived from traversal-style approaches and used to process structured text into a single contiguous sequence of tokens. They compare their results on four types of NLU task such as natural language inference, question answering, semantic similarity and text classification. They resulted improvement on 9 out of 12 tasks they studied. They explain their framework. Their training procedure consists of two steps whish are learning a high-capacity language model on a large corpus and a fine-tuning stage to adapt the model to a discriminative task. A single task-agnostic model is introduced in a framework using generative pre-training and discriminative pre-training. They achieve a strong natural language understanding in the experiments. Their model has the capability of processing long-range dependencies that is transferred to solve discriminative tasks summarized before (Radford et al.).

ELECTRA is an alternative to models like BERT which uses masked language modeling (MLM) as pre-training method. Clark et.al in their article on "Electra: Pre-Training Text Encoders as Discriminators Rather Than Generators" proposes a sample-efficient pre-training task called replaced token detection. It is a proposal that differs from BERT by not masking the text but corrupting some tokens. BERT works as corrupting the input by replacing some tokens with "MASK" tag and then training a model to reconstruct the original tokens. The authors' approach is to corrupt input by replacing some tokens with reasonable alternatives sampled from a small generator network. Their new approach performs better than masked language models (BERT) using the same model size, data and compute. The gains are particularly strong for small models and works well. Their approach trains two neural networks, a generator G and a discriminator D. They evaluate on the General Language Understanding Evaluation (GLUE) benchmark and Stanford Question Answering (SQuAD) dataset. Exact-Match (EM) and F1 Scores are used as evaluation metrics of. For most experiments they pretrain on the same data as BERT, consisting 3.3 Billion tokens from Wikipedia and BooksCorpus. They offer a new way of improving efficiency that is sharing weights between generator and discriminator.

## 2.1. Turkish morphology and NLP

Natural language processing, NLP, is a subfield of computer science in which the main concern is to understand human language in terms of computer understanding and process and analyze large amounts of natural language data. This is the theoretical definition of natural language processing. There are real world examples which are very common and efficiently solve many humane problems. Language comprehension, tagging such as asking Alexa to order something from Amazon. Machine translation such as using Google Translate to translate from German to English. Text Generation, summarization is another example so Google news showing a summary of a news article. Sentiment analysis is used where it is impossible to read and analysis the positivity or negativity of large number of writings. One of the most important tasks of NLP is named-entity recognition which is used in recommendation systems or voice assistants. In this thesis, I focus on NER for Turkish particularly.

If we want to apply specific rules or methodologies for a particular problem in NLP, we need to know the morphological structure of that language. Although there are models offered independently from the language, knowing the morphological structure and analyze those features increases the success of that particular task. So, Turkish is an agglutinating language meaning that the words in Turkish can be formed by long concatenation of morphemes.

Turkish is a suffixing language, excluding some borrowed derivational prefixes. Root forms are attached with derivational suffixes. In terms of inflectional morphology they are not as productive as inflectional morphology. (Çöltekin 2010) This makes this language relatively complex in terms of morphology. This is the reason why it needs special preprocessing like stemming or lemmatization. Çöltekin in his article, (Çöltekin 2014) explained the reasons of complexity of Turkish morphology.

## 2.2. Named Entity Recognition

In this thesis, main aim is to implement a NER tool. This tool will be an important part of the mobile travel assistant that will be an end product. This is basically a practical implementation of an ongoing research on NER. This can be considered as a real-life use case to show how important the NER task is. If we want our end product to be highly effective in terms of natural language understanding, we want it to achieve highest evaluation scores. Eventually, the system that incorporates our implemented NER implementation, would have an automated machine learning system which should learn from its own right or wrong labeling. We already defined what NER is now it is time to analyze NER with its entity types, how it is represented and how it is evaluated.

### 2.2.1. Entity types

In Sixth and Seventh Message Understanding Conferences (MUC)(Grishman et al. 1996) NER is initially defined to recognize fields of collectively known as ENAMEX: “persons”, “locations” and “organizations”. If these entities are expressions like “date” and “time” information, they are called TIMEX types. If they are numerical expressions like “money” and “percentage” quantities, then they are called NUMEX entities.

Depending on the application, additional types of entities can also be introduced such as proteins, medicines, etc., in medical text or particle names in quantum physics text.

### 2.2.2. Representation

Representation notion is needed mostly when there are more than one word in an entity. If an organizational entity has two names in it, for example names of sports clubs or name of governmental locations or offices, we need to mark the beginning and the end of those named entities. That is why representation is crucial and the choice of representation can affect the performance of NER systems.

Simply, named entities can use the raw tags by just tagging with its type. However, this is too simple to annotate sequential entities. IOB2 representation is commonly used to deal with 2 or more-word entities. It is also known as BIO. “B-” prefix is short for beginning and represents the beginning of the entity. Prefix “I-” is short for inside and represents the intermediary entities. If a word does not represent an entity, it is tagged with “O-” prefix which is short for outside.

BIO representation schema does not let us to tag the last word of named entity sequence. BILOU is another representation scheme that marks not only the beginning, inside or outside of a named-entity, but also the last token using the “L-” prefix. Additionally, U prefix to identifying the unit length named-entities with a “U-” prefix.

### 2.2.3. Evaluation

While training a NER system, generally token level performance of the model is evaluated. Besides of the general approach, there are other evaluation metrics which takes into evaluation at a full named-entity level. There are some possible scenarios that occur during evaluation and testing. These scenarios may not be well covered with the metrics in hand which are generally used. David Batista on his blog post named ‘Named-Entity evaluation metrics based on entity-level’(Batista 2021) discusses this issue and experiments those evaluation metrics. The necessity for different approaches arises from different scenarios as below:

- Surface string and entity type match. (True Positive)
- System hypothesized an entity (False Positive)
- System misses an entity (False Negative)

Considering these scenarios, CoNLL-2003(Tjong Kim Sang et al. 2003), Computational Natural Language Learning measures the performance of the system by computing precision, recall and f1-scores for each named-entity type. The percentage of entities found by the system correct is defined as precision. The percentage of entities that are in the corpus and found by the system is defined as recall. According to CoNLL-2003 a named entity is correct if it is an exact match of the entity in data file.

Those scenarios below are discarded in CoNLL-2003 format.

- System assigns the wrong entity type
- System gets the boundaries of the word wrong
- System gets the boundaries and entity type wrong

MUC: Message Understanding Conference in evaluation considering different categories(Chinchor et al. 1993) introduces more detailed metrics that covers most of the scenarios above mentioned like partial matching.

Different ways to measure precision recall and F1-score are introduced in International Workshop on Semantic Evaluation called SemEval. Correct and incorrect tokens are defined and measured differently. Partial and missed tokens are taken into consideration on calculations.

In these experiments, a Python framework for sequence labeling evaluation called Seqeval is used. This library measures the performance of the system according to the CoNLL-2003 standards. This library can evaluate the performance of tasks such as part-of-speech tagging, semantic role labeling and named-entity recognition task for this thesis. It supports most of the representation schemes but because our data is tagged with IOB2 representation scheme, it is suitable for this format, too.

Weighted average of the precision and recall is interpreted as F1 score. F1 score is at its highest when it reaches to 1, is at its lowest when it is at 0. Hence it is difficult to compare results with two different measures of precision and recall, these are combined into a single number represented with F letter. F1 score is not computed just using a simple arithmetic average but harmonic mean. Precision and recall affect the score relatively equal. The formula for the F1 score is:

Equation 2.1 $$F1 = 2 * (precision * recall) / (precision + recall)$$

## 2.3. Data Preparation Techniques

In this thesis 4 different datasets are utilized. These datasets are preprocessed and tagged automatically or manually. Universities or individual researchers gather and tag dataset for NLP community and make those datasets publicly available. In this thesis, some data preparation techniques are applied to the dataset before as I explained below. These methods had been applied to these datasets before this thesis.

### 2.3.1. Normalization

Normalization is a method that converts a list of terms into a uniform order. This is helpful when writing texts for further processing. Through adapting words to a common format, other processes would work with the data and do not have to deal with issues that might disrupt the procedure.

### 2.3.2. Tokenization

We need to tokenize the sentences to be able to recognize the entities. So, tokenization is a common and important task in NER. In order to analyze the sentence in terms of entities, we need to separate those sentences into smaller units called tokens. These tokens can be words, characters or sub-words. In this thesis only word-level tokenized datasets are used for NER task. Each word in a sentence is labeled. Sentences are separated from each other.

### 2.3.3. Stop words

Stop words typically apply to the most frequent words in a language, there is no one standardized stop word list used by all of the natural language processing tools. Removing stop words in NER raises the performance rate and affects the ranking, so stop words are generally omitted from the datasets.

### 2.3.4. Stemming and lemmatization

Stemming and lemmatization is to get rid of the inflectional groups and reach to a common base form. Although they may be used interchangeably, they have a nuance in the processes of stemming and lemmatization. Stemming is to reduce the inflected part and directly reach to the word stem. Removal of derivational suffixes helps us to find the root form most of the time. Stemming does not care about morphological root. However, lemmatization refers to process of finding the root of the word by morphological analysis and by the use of some vocabulary. The root form is called “lemma” in lemmatization.

## 2.4. Transfer Learning

NLP is a powerful tool in theory but when it is applied to real world, it suffers from data deficit and poor model generalization. Data deficit is the result of not having enough labeled data or domain specific data. Even the data exists, the usage and task may not be suitable for data. This results with the models that does not solve the aimed problems. Transfer learning models experienced in this thesis solved this problem by allowing us to take a predefined model of a task and use it for others.

Language model pretraining, known as transfer learning, has proven to be effective for improving most of the NLP token level tasks such as named entity recognition and question answering. It forms the basis for models like Bidirectional Encoder Representations from Transformers (BERT) and Embeddings from language models (ELMo). Before moving to the definition of transfer learning, we need to describe the notations and definitions used to form a formula.

In terms of applying pretrained language representations to downstream tasks two strategies are used: feature-based and fine tuning. ELMo (Peters et al. 2018) uses task specific architectures including pretrained representations as additional features. OpenAI GPT (Radford et al.) achieves the down streaming task by fine-tuning all pretrained parameters. The two approaches use unidirectional language models in order to learn general language representations.

### 2.4.1. Notations and Definitions

In this thesis, $\mathcal{D}$ Domain consists of a feature space $\mathcal{X}$ and a marginal probability distribution $\mathcal{P}(\mathcal{X})$, where $X = \{x_1, \dots, x_n\} \in \mathcal{X}$. Given our specific domain, which is travel in our case, $\mathcal{D} = \mathcal{X}, P(X)$, our task consists of two components: a label space $\mathcal{Y}$ and an objective predictive function $\mathfrak{f}(.)$ which is not observed buy learned from the training data, which consists of pairs $\{x_i, y_i\}$ where $x_i \in \mathcal{X}$ *and* $y_i \in \mathcal{Y}$. The function $\mathfrak{f}(.)$ is used to predict the label $\mathfrak{f}(x)$ of a new instance of $x$.

According to (Pan et al. 2010) transfer learning is defined as following:

Given a source domain $\mathcal{D}_S$ and learning task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, transfer learning aims to help improve the learning of the target predictive function $\mathfrak{f}_T(.)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{T}_S$ or $\mathcal{T}_S \neq \mathcal{T}_T$.

### 2.4.2. Categorization of transfer learning techniques

We can categorize transfer learning into three in terms of their related areas, availability of source domain labels and target domain labels and usage tasks.

Inductive Transfer Learning in which the target task is different from the source task. In this type of transfer learning difference or similarity of source and target domains are not taken into account. We need some amount of labeled data in target domain in order to induce an objective predictive model $\mathfrak{f}_T(.)$

Transductive Transfer Learning in which source and target tasks should be same. However, source and target domains should be different. In this case, large number of labels are available in the source domain. No labeled data is available in the target domain.

Unsupervised transfer learning in which the target task is different from the source task. There is no labeled data for source and target domains. The relationship between different types of transfer learning and the related areas are summarized in **Figure 2.1**

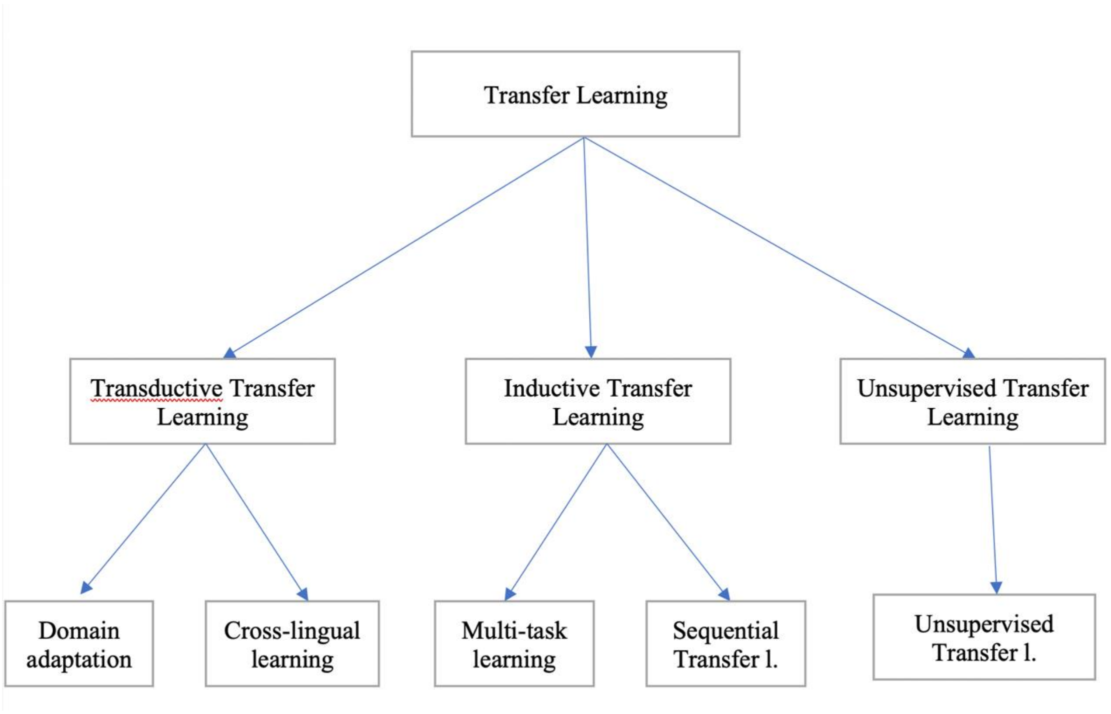


**Figure 2.1**. Types of Transfer Learning

### 2.4.3. Transfer learning models

After analyzing the transfer learning types, we need to analyze some transfer learning models in terms of how they perform in travel domain. In our case, there is some available labeled data. But this data is not domain specific. As explained in data preparation part of this thesis, data is taken from a Wikipedia Turkish dump in general topics and labeled. In this case, our type of transfer learning best fits into transductive transfer learning.

Pre-training language representations refers to training a general-purpose model and use that model for other tasks. The concept of pre-training is inspired from transferring knowledge between people. We transfer and use our gained knowledge and try to understand new knowledge.

For this thesis, we use different pre-trained language representation models of transfer learning to analyze the performance in travel domain. Models like BERT and ELECTRA are trained as a general-purpose "language understanding" model on a large text corpus (like Wikipedia), and then can be used for down streaming NLP tasks that we care about like token classification and question answering.

Pre-trained representations can be categorized in terms of their context dependencies such as context-free or contextual. Word2vec or GloVe can be given examples for context-free models. They generally generate a single "word embedding" for each word in the vocabulary. However, contextual models take the other words in the sentence into account and generate a representation for each word.

Contextual representations are also divided into unidirectional or bidirectional. Pretraining contextual representations such as Semi-supervised Sequence Learning, Generative Pre-Training, ELMo, and ULMFit are categorized as unidirectional or shallowly bidirectional but also these models prepared the environment for BERT. Unidirectional means that each word is only contextualized using the words to one side only. These two models below are contextual models that will be used in the experiments

BERT (from Google) released with the paper BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding by Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova.(Devlin et al. 2019)

ELECTRA (from Google Research/Stanford University) released with the paper ELECTRA: Pre-training text encoders as discriminators rather than generators by Kevin Clark, Minh-Thang Luong, Quoc V. Le, Christopher D. Manning.(Clark et al. 2020)

#### 2.4.4. BERT

A new language model representation is introduced in 2018 called BERT which stands for Bidirectional Encoder Representations from Transformers. BERT is also commonly known as a masked language model. (MLM) This model differs from other models introduced until 2018 with its being bidirectional. Bidirectionality means while pretraining the input data it takes into account both the left and right context in all layers. Instead of using the standard left-to-right prediction of the next word as the target task, BERT proposes two new tasks.

BERT outperforms previous methods because it is the first unsupervised, deeply bidirectional system for pre-training NLP. Unsupervised means that BERT was trained using only a plain text corpus which is plain text data publicly available on the web in many languages. BERT is a fine-tuning based and encoder-based method. So that pre-trained BERT model can be fine-tuned adding only one additional output layer.

BERT represents a word using both its left and right context starting from the very bottom of a deep neural network, so it is deeply bidirectional. In the input word sequence of this model, 15% of the words are randomly masked and the task is to predict what they are. What we see is that, unlike previous models, BERT can predict these words from both directions — not just left-to-right or right-to-left. In addition, in terms of model input, BERT WordPiece embedding is used instead of standard word embedding. That's because BERT, by default, doesn't use words as tokens but word pieces. From here, a position embedding and a segment embedding are added. In order to learn relationships between sentences, they also train on a simple task which can be generated from any monolingual corpus. Using BERT has two stages: Pre-training and fine-tuning. The other important aspect of BERT is that it can be adapted to many types of NLP tasks very easily.

### 2.4.5. ELECTRA

We learned that BERT is one of the Masked Language Modeling pre-training methods. BERT functions as corrupting the input by replacing some tokens with “MASK” and then training a model to reconstruct the original tokens. This model has advantage of producing good results, but it requires large amounts of computation to generate effective results.

ELECTRA is a new method for self-supervised language representation learning introduced in 2019. It is proposed as an alternative to BERT. It needs relatively little compute to train transformers network. It uses replaced token detection method which is claimed to be more sample-efficient. It works like corrupting the input by replacing some tokens with alternatives from a small generator network. It does not mask the input like BERT does. Instead of predicting the originals of the corrupted tokens, it trains a discriminative model which predicts if the token is replaced or not. More details about ELECTRA can be found in the ICLR paper (Clark et al. 2020) or in the official ELECTRA repository (Research 2020) on GitHub.

# 3. MATERIAL AND METHOD

## 3.1. Data Gathering

Four types of data are used for named entity recognition. The reason to use different datasets is to analyze how the model success scores are affected with the change of data.

The first dataset is called TWNERTC (Sahin et al. 2017). They used Wikipedia dumps and google Freebase (depreciated as of now) to automatically annotate named entities and categories of sentences. Turkish Wikipedia dumps are utilized as the text source and Freebase is used to construct a large-scale gazetteer to map fine-grained types to entities. Domain information provided by Freebase is utilized to develop domain independent and domain dependent methodologies to overcome noisy and ambiguous data.

The second source of data is called İTÜ NER (Şeker et al. 2017). It is acquired from ITÜ NLP Service (Eryiğit 2014) with the permission of usage and it is licensed under Creative Commons. This dataset mainly consists of 500K words collected from newspaper articles and is annotated only for ENAMEX types. In their work, Şeker et.al tried to extend the NER types with TIMEX and NUMEX expressions

The third dataset is called WikiANN which is a preprocessed balanced dataset. There are three types of files named train.txt, dev.txt and test.txt on the website. There are total of six labels such as B-LOC, B-ORG, B-PER, I-LOC, I-ORG, I-PER and O for not labeled ones. There's a token/label pair for each line, delimited by a space. An empty line denotes a space.

The third dataset is mainly generated with gazetteers. In ("Schweter Wikiann Webpage") and (Şeker et al. 2017) the authors used two kind of gazetteers called base and generator gazetteers. In this thesis, same technique is utilized to create annotated travel data to add more layers to train model. Large gazetteers with thousands of tokens are compiled mostly using Wikipedia as the data source. These gazetteers are explained as below in **Table 3.1**

**Table 3.1.** Number of distinct tokens in gazetteers

| **gazetteer** | **Number of tokens** |
|---|---|
| Airports in Turkey | 104 different airports ~220 tokens |
| Airports in the world | 12669 different airports ~16000 tokens |
| Airlines in the world | 6163 different airlines ~10000 tokens |
| Largest cities in Turkey | 581 different cities ~700 tokens |
| Countries of the world | 228 different countries ~500 tokens |

## 3.2. Data Preparation

NER experiments are completed with data that were pre-processed and prepared for training and fine-tuning. Although it is preprocessed, for every transfer learning model the data should be tagged or formatted. For instance, BERT expects input data in a specific format. Dataset should be tokenized, and some special tokens should be added to mark the beginning “CLS” and separation/end of the sentences “SEP”. The sentences should be vectorized and all the vectors should be of the same size. That is why a common sentence length should be defined. This length will be decided by the length of longest sentence, which we will have to calculate. The sentences should be padded to achieve common length. Then the actual tokens and the padded ones should be differentiated between with the help of "Attention Masks". If we shortly describe what is meant with attention masks looking at the hugging face documentation: The attention mask is a binary tensor indicating the position of the padded indices so that the model does not attend to them. For the BertTokenizer, 1 indicate a value that should be attended to while 0 indicate a padded value.

The data coming from gazetteers, automatically annotated Turkish dataset, WikiANN dataset and İTÜ NER dataset are used in order to prepare the feature vectors for train test and validation instances.

Community-driven BERTurk (Schweter 2020a) is used as the base BERT models. A more detailed explanation and implementation details can be found under this GitHub link (Schweter 2020b). Datasets used for pretraining and evaluation are contributed with Turkish NLP community. The current version of the model is trained on Turkish OSCAR corpus (OSCAR), a recent Wikipedia dump, various OPUS corpora and a special corpus provided by Kemal Oflazer. The final training corpus has a size of 35GB which means millions of tokens. Cased and uncased models are trained using Google's TensorFlow Research Cloud (TFRC) on a TPU v3-8. DBMDZ web site stores all the trained models for public use. The following models are available:

- BERTurk models with 32k vocabulary: dbmdz/bert-base-turkish
  cased and dbmdz/bert-base-turkish-uncased

- BERTurk models with 128k vocabulary: dbmdz/bert-base-turkish-128k-
  cased and dbmdz/bert-base-turkish-128k-uncased

- ELECTRA small and base cased models (discriminator): dbmdz/electra-small-
  turkish-cased-discriminator and dbmdz/electra-base-turkish-cased-discriminator

Small and base ELECTRA models for Turkish, trained on the same data as BERTurk, is used as base models. Both small and base ELECTRA models are trained for one million steps and do intrinsic and extrinsic evaluations. Extrinsic evaluations are done for PoS tagging and NER. During pre-training checkpoints were written every 100k steps. For the final evaluation, 50 experiments on each downstream task were performed to select the best and final checkpoint for the model release. More details can be found here under (Schweter 2020b).

# 4. RESULTS AND DISCUSSION

**Table 4.1** shows results of first set of NER experiments using BERT models: Bert-base-multilingual-cased, bert-base-turkish-128k-cased bert-base-turkish-cased. In these experiments, a Python framework for sequence labeling evaluation called Seqeval is used. This library measures the performance of the system as CoNLL-2003 format defined before.

All of these pretrained models are downloaded from the HuggingFace Transformers official page. (Hugging). Community-uploaded models can be found from a curated list given in the official website. During the fine tuning, total path should be written as ‘dbmdz/bert-base-turkish-cased’. For the sake of shortness, only the name of the model will be written in tables for the model names.

When we look at the initial NER with BERT experiments, we can see that the data quality is highly effective in the success scores. The highest score achieved is 0.92 with the WikiANN data which is well organized and labeled. It is also homogenous in terms of labels. That is the reason even though it is less fine-tuned in terms of epoch counts, it achieves more success.

The first two rows of the **Table 4.1** shows the f1 scores of ITU NER data with two different models. It achieves the most with the first BERT model. Even though increasing the vector size to 128 does not affect the result that much as expected.

On the other hand, TWNERTC data originally has more than 100 labels, which means that number of columns while conducting fine tuning but, as we can see from the results even if we increase the labels, it affects negatively in the success rates. The last two rows of the **Table 4.1** shows us that increasing epoch count does not change the results dramatically. Increasing epoch from 1 to 3 only changes the scores 0.02 percent.

**Table 4.1.** BERT experiments-different models, various epochs

| Model | Dataset | Batch | Epoch | Loss | Precision | Recall | F1 |
|---|---|---|---|---|---|---|---|
| bert-base-multilingual-cased | İTÜNER | 16 | 10 | 0.0710 | 0.6875 | 0.75 | 0.71 |
| bert-base-turkish-128k-cased | İTÜNER | 16 | 10 | 0.0554 | 0.6250 | 0.79 | 0.70 |
| bert-base-turkish-cased | WikiANN | 16 | 3 | 0.1209 | 0.9213 | 0.93 | **0.92** |
| bert-base-turkish-cased | TWNERTC | 16 | 1 | 0.2611 | 0.6781 | 0.65 | 0.66 |
| bert-base-turkish-cased | TWNERTC | 16 | 3 | 0.2727 | 0.6882 | 0.68 | 0.68 |

**Table 4.2** shows us the F1 scores of controlled fine-tuning experiments using the same models, batch sizs and epoch counts through three different types of data we have

been using. These results show us the scores without augmented data injected. WikiANN data performs the best.

**Table 4.2.** BERT same models, epochs, batches without augmented data

| **Model** | **Dataset** | **Batch** | **Epoch** | **Loss** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|---|---|---|
| bert-base-turkish-cased | İTÜNER | 16 | 3 | 0.0882 | 0.4670 | 0.5033 | 0.4845 |
| bert-base-turkish-cased | WikiANN | 16 | 3 | 0.1209 | 0.9213 | 0.9308 | **0.9260** |
| bert-base-turkish-cased | TWNERTC | 16 | 3 | 0.2548 | 0.6781 | 0.6556 | 0.6637 |

**Table 4.3** shows us augmented data injected version of the Table 4.2. The batch and epoch counts are hold the same. The augmented data is prepared for only intention of Flight Search in travel domain. There are 6 types of sentences prepared to understand this intent. In addition to the labels used in the datasets, some additional labels are also added to the augmented data such as B-NUM, I-NUM for representing numbers and B-DATE, I-DATE for representing dates. These generated sentences and the token classification implementation in python are submitted to this public GitHub repository. (https://github.com/denizgul/token-classification)

If we discuss the results, the model fined-tuned with İTÜNER data achieved 0.48 score without the augmented data in table **Table 4.2**, it achieves 0.67 f1 score with the augmented data. This proves us that we can perform better results with data injection. This is very important because it means we can create domain specific data and increase our model’s success rate. But also, we should keep in mind that when the success rates come close to 0.90’s, the effect of the data injection lowers. We can prove it such as, the effect to the first model is higher than to the second WikiANN finetuned model. If the success scores the curve of the success scores and time flattens as it comes closer to the state-of-the-art results.

**Table 4.3.** BERT augmented data injected

| **Model** | **Dataset** | **Batch** | **Epoch** | **Loss** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|---|---|---|
| bert-base-turkish-cased | İTÜNER | 16 | 3 | 0.2178 | 0.7159 | 0.6319 | 0.6713 |
| bert-base-turkish-cased | WikiANN | 16 | 3 | 0.1115 | 0.9280 | 0.9347 | **0.9313** |
| bert-base-turkish-cased | TWNERTC | 16 | 3 | 0.2611 | 0.6801 | 0.6445 | 0.6692 |

Two most successful dataset is selected from the above results and evaluated with the same test and eval set. The scores are written below in **Table 4.4**

While testing the model with the same test data, it is important to change the labels according to the model. For example, in WIKIANN data, the model is fine-tuned according to the labels that is size of 10, however in İTÜNER data the labels' size is 17. If you happen to use those models, the transformers will give an error on wrong vector size. Also, the labels should refer to the same token. For example, the location is referenced as B-LOC and I-LOC in WIKIANN data, whereas it is labeled as B-LOCATION in İTÜNER data. Non-uniformity in labels may cause potential errors while fine tuning.

**Table 4.4.** BERT same finetuned model, test and eval data

| Dataset | E.Loss | E.Precision | E.Recall | E.F1 | T.Loss | T.Precision | T.Recall | T.F1 |
|---|---|---|---|---|---|---|---|---|
| İTÜNER | 1.4279 | 0.5537 | 0.4871 | 0.5183 | 1.4222 | 0.5512 | 0.4861 | 0.5166 |
| WikiANN | 0.1138 | 0.9296 | 0.9395 | 0.9346 | 0.1115 | 0.9280 | 0.9347 | 0.9313 |

E: Eval T: Test

Experiments show that ELECTRA is more efficient than MLM. The reason is that ELECTRA is defined over all input tokens rather than just the small subset that was masked out.

The following **Table 4.5** is created as the result of the NER experiments using the above ELECTRA models. It shows the results which are experimented on the same data and compute power. The situation does not change in those results as well. The most successful results come from the WikiANN data. The worst success is achieved by TWNERTC automatically annotated data. ELECTRA achieves almost same with BERT with the highest score 0.91. ELECTRA is a faster way of BERT in terms of fine tuning as suggested. The worst success is achieved by TWNERTC automatically annotated data. ELECTRA achieves almost same with BERT with the highest score 0.91. ELECTRA is a faster way of BERT in terms of fine tuning as suggested.

**Table 4.5.** ELECTRA experiments-different epochs-different models

| Model | DataSet | Batch | Epoch | Loss | Precision | Recall | F1 |
|---|---|---|---|---|---|---|---|
| electra-base-turkish-cased-discriminator | İTÜNER | 8 | 10 | 0.06 | 0.62 | 0.79 | 0.70 |
| electra-small-turkish-cased-discriminator | İTÜNER | 8 | 10 | 0.04 | 0.72 | 0.81 | 0.76 |
| electra-small-turkish-cased-discriminator | WikiANN | 8 | 10 | 0.18 | 0.90 | 0.91 | **0.91** |
| electra-small-turkish-cased-discriminator | TWNERTC | 8 | 3 | 0.29 | 0.64 | 0.63 | 0.63 |

**Table 4.6** shows the experiments with the same batch size:16 and epoch count 3 for the sake of standardization. One thing that catches attention here is WikiANN data performs better when we increase batch size to 16 and decrease the epoch count to 3.

**Table 4.6.** ELECTRA experiments-same models-same epochs

| **Model** | **DataSet** | **Batch** | **Epoch** | **Loss** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|---|---|---|
| electra-small-turkish-cased-discriminator | İTÜNER | 16 | 3 | 0.0873 | 0.4184 | 0.4605 | 0.4384 |
| electra-small-turkish-cased-discriminator | WikiANN | 16 | 3 | 0.1163 | 0.9213 | 0.9318 | **0.9265** |
| electra-small-turkish-cased-discriminator | TWNERTC | 16 | 3 | 0.2513 | 0.6799 | 0.6433 | 0.6622 |

**Table 4.7.** ELECTRA experiments with augmented data

| **Model** | **DataSet** | **Batch** | **Epoch** | **Loss** | **Precision** | **Recall** | **F1** |
|---|---|---|---|---|---|---|---|
| electra-small-turkish-cased-discriminator | İTÜNER | 16 | 3 | 0.1868 | 0.6060 | 0.6006 | 0.6033 |
| electra-small-turkish-cased-discriminator | WikiANN | 16 | 3 | 0.1088 | 0.9304 | 0.9397 | **0.9350** |
| electra-small-turkish-cased-discriminator | TWNERTC | 16 | 3 | 0.2501 | 0.6834 | 0.6433 | 0.6703 |

**Table 4.8.** ELECTRA same finetuned model, test and eval data

| **Dataset** | **E.Loss** | **E.Precision** | **E.Recall** | **E.F1** | **T.Loss** | **T.Precision** | **T.Recall** | **T.F1** |
|---|---|---|---|---|---|---|---|---|
| İTÜNER | 1.1908 | 0.5862 | 0.5561 | 0.5708 | 1.1840 | 0.5816 | 0.5553 | 0.5681 |
| WikiANN | 0.1201 | 0.9198 | 0.9354 | 0.9301 | 0.1213 | 0.9219 | 0.9328 | **0.9273** |

In these NER experiments only transfer learning methods are used. These transfer learning models are BERT and ELECTRA which performs different according to the data. Four different data are collected from different sources. The aim of using various data types is to prove how data quality affects even with the transfer learning models. Some may argue that with the transfer learning the importance of data may lessen.

However, when it comes to fine tuning, the importance of data quality for the specific tasks still remains valid. When we look at the results, we can see that data called WikiANN is performing higher because it is more accurate and balanced.

TWNERTC is known as being labeled automatically using unsupervised methods and using large-scale gazetteers. TWNERTC was expected to perform higher looking at the amount of model and data used while creating the data. However, when we look at the results, it performs the worst f1 score among the others. TWNERTC has 48 different labels in the version I used. As a result, increasing model size and label count may decrease the model accuracy. This proves that more data and more labels may not succeed more. The accuracy of labels is open discussion.

The official code implementation of BERTURK suggests that the highest success rates using BERTurk and ELECTRA for NER task is 0.9555 and 0.9567 respectively. The experiments performed in this thesis could not achieve those results. The highest success rates achieved here for BERT and ELECTRA is 0.92 and 0.91 respectively.

Augmented data is created with appending the most successful data and generated data together and created a new corpus. The size of the WikiANN data is around 2.5 MB and the generated data with all the gazetteers explained above is close to WikiANN, is around 2.6 MB. For the sake of controlled experiments, sizes are hold close so that it will not affect the results. However, the generated data did not affect the results dramatically.

Augmented data is generated with 6 type of sentences. For example, I want a flight ticket from Antalya to İstanbul for the next Tuesday. (Antalya’dan İstanbul’a haftaya Salı bir kişilik uçak bileti istiyorum.) All the sentences start with from location to location and continues with a date tag and with the numeric value. The data is labeled accordingly. For example, ‘haftaya salı’ is labeled as B-DATE, I-DATE respectively. ‘Bir kişilik’ is labeled as B-NUM, O respectively.

ELECTRA and BERT almost perform the same, BERT performs better on the same data with 0.01 points difference. Epoch count may affect the end result notably, it may generally increase the success rate using other deep learning methodologies. As we can see from the results using TWNERTC data in table 3.2 Bert Experiments, the rates change only 0.02 with the increase of epoch count. TWNERTC data takes 100 longer even with 3 epochs. Because there are more 7 times more labels than the other datasets. The size of the data is also much higher than WikiANN data. Even though the sizes and labels are higher it performs the worst. ELECTRA is shortest in terms of training evaluation and test tasks on NER.

Increasing number of epochs can be experimented with bigger computes. In these NER Experiments 16 GB RAM, intel core i7 processor with NVIDIA GeForce GTX 1660 Ti 6GB machine is used. Total of 25 experiments conducted but 11 of them are written in the tables. The other 14 experiments were unsuccessful because of computation is not enough most of the cases.

Token classification task is already implemented on transformers home page. For our NER implementation those were used as the starting point. The data preprocessing part is added according to the data acquired from various sources

# 5. CONCLUSION

Natural language processing is a wide area in which research is still going on. Today, there are multiple tasks which still carry importance like question answering, classification and named entity recognition (NER). In this thesis, some of the previous works that achieve state of the art results on NER task are reviewed. Almost all of those works in Turkish are established using methods other than transfer learning.

In this thesis, transfer learning methods are explained and experimented on NER task. Commonly used transfer learning methods such as ELECTRA and GOOGLE BERT are compared with each other. Four different data acquired from different sources are used in experiments and discussed the results. When the results compared each other, we can see the importance of data quality still remains valid. When we fine tune without changing the model with an unbalanced data, negative transfer occurs meaning we are better off when we do not do transfer learning. For the NER task automatic labeling did not offer state off the art results' success. Automatic labeling shows high performance with the validation data, when we apply those models to other models data, it generates low performance which is impractical. For a given specific domain, we can increase the success rates by data injection which is specifically prepared for the specific intentions.

In the future, some important research problems need to be taken into account. Researchers are experimenting using various labeled data and in some cases the learning may perform badly than the experimented results. In more detail, we expect to see better results when we transfer knowledge from model to our model. When the source and target domains do not relate to each other in some sense, it means there is not knowledge to be transferred. In that case learning can be negative. Negative transfer should be avoided, and more research should be performed on how to avoid it.

## 6. REFERENCES

Af, A., and Ak, M.D. Zemberek, an open source NLP framework for Turkic Languages. 8.

Batista, D. 2021. *Named-Entity evaluation metrics based on entity-level*, 2021. http://www.davidsbatista.net/blog/2018/05/09/Named_Entity_Evaluation/ [accessed Jan. 12, 2021].

Çelikkaya, G., and Eryiğit, G. 2014. A Mobile Assistant for Turkish. *Türkiye Bilişim Vakfi Bilgi. Bilim. Ve Mühendisliği Derg.*, 7: 38–42.

Chinchor, N., and Sundheim, B. 1993. MUC-5 Evaluation Metrics. Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993, p.

Clark, K., Luong, M.-T., Le, Q.V., and Manning, C.D. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *ArXiv200310555 Cs*,.

Çöltekin, Ç. 2010. A Freely Available Morphological Analyzer for Turkish. Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10), (Valletta, Malta: European Language Resources Association (ELRA)), p.

Çöltekin, Ç. 2014. A set of open source tools for Turkish natural language processing. Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), (Reykjavik, Iceland: European Language Resources Association (ELRA)), pp. 1079–1086.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv181004805 Cs*,.

Eryiğit, G. 2014. ITU Turkish NLP Web Service. Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics, (Gothenburg, Sweden: Association for Computational Linguistics), pp. 1–4.

Eryigit, G. 2017. Use of NLP Techniques for an Enhanced Mobile Personal Assistant: The Case of Turkish. *Int. J. Intell. Syst. Appl. Eng.*, 3: 94–104.

Eryiğit, G., Nivre, J., and Oflazer, K. 2008. Dependency Parsing of Turkish. *Comput. Linguist.*, 34: 357–389.

Grishman, R., and Sundheim, B. 1996. Message Understanding Conference- 6: A Brief History. COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics, p.

Gunes, A., and TantuG, A.C. 2018. Turkish named entity recognition with deep learning. 2018 26th Signal Processing and Communications Applications Conference (SIU), (Izmir: IEEE), pp. 1–4.

Hugging, F. https://huggingface.co/transformers/pretrained_models.html [accessed Jan. 12, 2021].

Küçük, D., and Yazıcı, A. 2009. Named Entity Recognition Experiments on Turkish Texts. Flexible Query Answering Systems, T. Andreasen, R.R. Yager, H. Bulskov, H. Christiansen, and H.L. Larsen, eds. (Berlin, Heidelberg: Springer Berlin Heidelberg), pp. 524–535.

Nadeau, D., and Sekine, S. 2007. A survey of named entity recognition and classification. *Lingvisticæ Investig.*, 30: 3–26.

Okur, E., Demir, H., and Özgür, A. 2016. Named Entity Recognition on Twitter for Turkish using Semi-supervised Learning with Word Embeddings. Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), (Portorož, Slovenia: European Language Resources Association (ELRA)), pp. 549–555.

OSCAR *OSCAR*. https://oscar-corpus.com/ [accessed Nov. 02, 2020].

Pan, S.J., and Yang, Q. 2010. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.*, 22: 1345–1359.

Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. 2018. Deep contextualized word representations. *ArXiv180205365 Cs*,.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving Language Understanding by Generative Pre-Training. 12.

Research, G. 2020. *GitHub-Electra*, 2020. https://github.com/google-research/electra [accessed Nov. 03, 2020].

Sahin, H.B., Tirkaz, C., Yildiz, E., Eren, M.T., and Sonmez, O. 2017. Automatically Annotated Turkish Corpus for Named Entity Recognition and Text Categorization using Large-Scale Gazetteers. *ArXiv170202363 Cs*,.

Schweter, S. 2020a. 2020. https://doi.org/10.5281/zenodo.3770924.

Schweter, S. 2020b. stefan-it/turkish-bert. Python.

Şeker, G.A., and Eryiğit, G. 2012. Initial Explorations on using CRFs for Turkish Named Entity Recognition. Proceedings of COLING 2012, (Mumbai, India: The COLING 2012 Organizing Committee), pp. 2459–2474.

Şeker, G.A., and Eryiğit, G. 2017. Extending a CRF-based named entity recognition model for Turkish well formed text and user generated content1. *Semantic Web*, 8: 625–642.

Sezer, T. 2017. TS Corpus Project: An online Turkish Dictionary and TS DIY Corpus. *Eur. J. Lang. Lit.*, 9: 18.

Sezer, T., and Sezer, B. 2013. TS Corpus: Herkes için Türkçe Derlem. p.

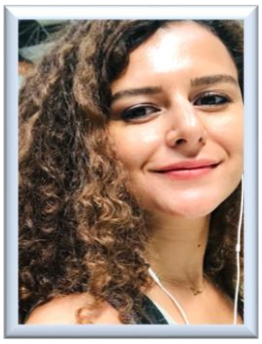
Tjong Kim Sang, E.F., and De Meulder, F. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, pp. 142–147.

2018. Turkish Natural Language Processing. Springer International Publishing,.

**RESUME**

**DENİZ GÜL ÖZCAN**

**denizgul@gmail.com**



## EDUCATION

| | |
|---|---|
| MSc<br>2018-2021 | Akdeniz University<br>Institute of Natural and Applied Sciences,<br>Computer Engineering, Antalya, Turkey |
| BSc<br>2008-2014 | Koc University<br>Faculty of Engineering,<br>Computer Engineering, İstanbul, Turkey |

## EMPLOYMENT

| | |
|---|---|
| 2020-Present | SSAT Turizm ve Ticaret Ltd. Şti İş Geliştirme Md,<br>Uzman Araştırma Görevlisi,<br>Antalya, Türkiye |
| 2019-2020 | ETS TUR, Uzman Yazılım Geliştiricisi,<br>Antalya, Türkiye |
| 2015-2019 | IATI-Aerobilet, B2C Takım Lideri Uzman Java Yaz. Geliştiricisi,<br>Antalya, Türkiye |
| 2012-2015 | IATI-Aerobilet, Android-IOS Yazılım Geliştiricisi,<br>Antalya- Türkiye |
| 2010-2012 | TEZTOUR, Java Yazılım Geliştirici,<br>Antalya, Türkiye |
| 2009-2010 | L-Mobile Solutions, .NET Developer,<br>Stuttgart, Almanya |
| 2008-2009 | Daisho Blacksmith, Java Yazılım Geliştiricisi,<br>Münih, Almanya |

## PUBLICATIONS

Ozcan, D. G. 2019. Smart Cities Performability, Cognition, Security, Bolum adı: Open Source Tools for Machine Learning with Big Data in Smart Cities, Uluşar Ü.D., Alturjman F, Springer, Ingilizce (Bilimsel Kitap) https://link.springer.com/chapter/10.1007/978-3-030-14718-1_8