ISTANBUL TECHNICAL UNIVERSITY
FACULTY OF COMPUTER AND
INFORMATICS

Automatic Term Extraction From Turkish Legal Texts

Graduation Project Final Report

Uzay Uysal
150180039

Department: Computer Engineering
Division: Computer Engineering

Advisor: Assoc. Prof. Dr. Gülşen Eryiğit

June 2022

Statement of Authenticity

I/we hereby declare that in this study

1. all the content influenced from external references are cited clearly and in detail,
2. and all the remaining sections, especially the theoretical studies and implemented software/hardware that constitute the fundamental essence of this study is originated by my/our individual authenticity.

İstanbul, Haziran 2022

Uzay Uysal

Automatic Term Extraction From Turkish Legal Texts

(SUMMARY)

Term extraction is an operation that aims to identify the core vocabulary of a specialized domain. This extraction operation plays an important role in many fields like translation, indexing, standardisation and localisation since the extracted terms essentially provide a summary of the text which can be directly translated into the corresponding terms in the target language. In the conventional methods this operation is done in two steps. Firstly, a terminologist picks candidate terms from the selected text to create a list of candidates and then as the second step, the candidate term list is shown to a domain expert to form the final list of validated terms. Since this method requires human intervention in both steps, it is both a challenging and a time-consuming task. Automatic term extraction aims to make this task easier, less time-consuming and possibly completely automated by using computer programs that take advantage of statistical and artificial intelligence based methods. Statistical methods generally use metrics like word and multi-word idiom frequencies, part-of-speech(POS) tagging and morphological analysis while artificial intelligence based methods use more advanced methods like word embeddings and neural networks. Even though there is plenty of research in the field, most of them are in more common languages like English and state-of-the-art methods still are not good enough to pick the terms automatically, instead they can produce a list of candidate terms. There are a few issues that hinders the development of an algorithm good enough to automate the whole process. One of them is the difficulty in deciding what is a term. Another one is that languages are too complex to make general statements on them and word frequencies are not enough to decide if a word is a term or not. Lastly, the lack of high quality multilingual datasets is an important issue for artificial intelligence based methods since these methods require huge amount of labeled data to get good results. There are only a few proper datasets available and they are mostly in English which obstructs the usage of supervised systems in less common languages.

In this project, I developed a whole pipeline that preprocesses and extracts candidate terms from Turkish legal documents. Since there is no former study in the field that deals with Turkish legal documents and no dataset has been provided for the project, I used a hybrid method on a custom dataset that uses both statistical methods and methods like word embeddings and co-training that take advantage of neural networks. The algorithm extracted the candidate terms from a corpus that is collected and compiled from Yargıtay Karar Arama interface using a web-scraping script and the resulting terms are compared to the terms in the T.C. Adalet Bakanlığı Hukuk Sözlüğü as a means to see how successful the algorithm is. The word embeddings generated by the neural network are saved to be used in the later studies. As the end result, it can be said that the algorithm can extract at least 40-45 words that are labeled as correct by T.C Adalet Bakanlığı Hukuk Sözlüğü by analyzing its top 100 guesses. It is not possible to know the actual correct extracted term count since there is not a labeled dataset. The algorithm also filters out words that are used very frequently in the Turkish language successfully which proves that it can be a useful tool to generate candidate terms from Turkish legal documents to be later inspected by human evaluators.

Hukuk Alanındaki Metinlerden Otomatik Terim Çıkarma

(ÖZET)

Terim çıkarma (term extraction) özünde spesifik bir alanı en iyi ifade eden kelimelerin çıkarılmasını amaçlayan bir operasyondur. Bu çıkarma işlemi, çıkarmanın yapıldığı metni özetlemesi ve çıkarılan kelimelerin terim olması sonucu başka dillerdeki karşılıklarına isabetli şekilde çevrilebildiği için diller arası çeviri yapma, standartlaştırma, yerelleştirme ve doküman listeleme gibi alanlarda önemli role sahiptir. Çıkarma işlemi geleneksel metotlarda iki adımda yapılır. İlk olarak terminoloji alanında uzman bir kimse seçilen metinden aday terimleri (candidate terms) seçer ve bunların bir listesini oluşturur. İkinci adımda ise bu liste metnin ait olduğu alanın uzmanı bir kimseye gösterilerek onaylanmış terim listesi oluşturulur. Bu metot iki adımda da insan yardımı gerektirdiği için zor ve zaman alıcı bir işlemdir. Otomatik terim çıkarma (automatic term extraction) bu işlemi daha kolay ve daha az zaman alıcı bir hale, hatta mümkünse tamamen otomatikleştirilmiş bir hale getirmek için istatistiksel verilerden ve yapay zeka modellerinden yararlanan bilgisayar programları kullanır. İstatistiksel verilerden yararlanan metotlar genelde kelime ve kelime grubu frekansları, cümlenin ögeleri ve morfolojik analiz gibi teknikleri kullanırken; yapay zeka tabanlı metotlar kelime temsilleri (word embedding) ve yapay sinir ağları gibi daha gelişmiş teknikleri kullanıyor. Alanda birçok araştırma olmasına rağmen, araştırmaların çoğu İngilizce ve İngilizce'ye yakın dillerde yapılmış ve güncel metotlar hala tamamen otomatik şekilde terimleri çıkaracak kadar iyi değiller. Bunun yerine genelde aday terim lisesi çıkarıyorlar. Bu kadar araştırma olmasına rağmen işlemi tamamen otomatikleştirecek algoritmaların geliştirilememesinin sebebi olarak bu işlemi yavaşlatan birkaç engel gösterilebilir. Bu engeller kısaca, terimleri tam olarak tanımlamanın zor olması, istatistiksel metotlar için dillerin kelime frekanslarıyla terimleri çıkarmak için fazla karmaşık olması ve yapay zeka tabanlı modeller için ise kaliteli veri kıtlığı olarak özetlenebilir. Yapay zeka tabanlı modeller genelde iyi sonuçlar almak için oldukça büyük veri kümelerine ihtiyaç duyuyorlar. Terim çıkarımı alanında şu anda var olan veri kümelerinin ise sayısı az ve bu veri kümeleri genellikle İngilizce dilindeler. Bu da gözetmeli öğrenme (supervised learning) kullanan modeller için sorun teşkil ediyor.

Bu projede, hukuk alanındaki Türkçe metinleri filtreleyip otomatik şekilde aday terimleri çıkaran bir algoritma geliştirdim. Daha önce Türkçe hukuk metinleri ile yapılan bir çalışma olmaması ve proje için herhangi bir veri kümesi sağlanmamasından dolayı, istatistiksel metotları, kelime temsilleri ve co-training metodunu kullanan karma bir algoritma geliştirdim. Geliştirilen algoritma Yargıtay Karar Arama internet sitesinden veri kazıma ile toplanan ve derlenen bir veri kümesi ile eğitildi ve sonuçlar T.C. Adalet Bakanlığı Hukuk Sözlüğü'ndeki terimler ile karşılaştırılarak algoritmanın genel performansı incelendi. Elde edilen veri temsilleri daha sonraki çalışmalarda kullanılmak üzere kaydedildi. Algoritmanın çıkardığı aday terimlerin sözlükteki terimlerle karşılaştırılması üzerine, algoritmanın tahmin ettiği en iyi 100 kelimeden en az 40-45 civarının doğru olduğu söylenebilir. Ancak etiketli bir veri kümesi olmadığı algoritmanın tam performansını ölçmek mümkün değil. Sonuç olarak algoritma Türkçede oldukça sık kullanılan sözcükleri başarılı şekilde elediği için insan değerlendiricilerin metinlerin tamamını incelemesi yerine sadece aday terimleri inceleyerek zaman kazanmalarını sağlayabilir.

Contents

1 Introduction and Project Summary

Terminologies are getting more and more important in our everyday life as the amount of knowledge increases rapidly with the advance of technology and science. For this reason, there has been a need for research in the field to augment terminology related tasks. Different approaches like using computational methods gave birth to new research fields. Automatic term extraction (ATE) is one of the fields that has taken a lot of attention in that sense which is also the subject of this project. ATE systems aim to extract terms from domain specific corpus in an automated manner using computational methods. Terms are essentially words or word groups that are specialized to a specific domain which refers to a specific concept in that domain. This property makes them a valuable resource when indexing documents, articles or any text in that kind since they provide a general idea about the concepts explained in the aforementioned texts. Therefore, the extraction of terms play a huge role in the advancement of fields like information retrieval as it can help with listing better results for user queries. Term extraction can also be useful with tasks like machine translation since word by word translation of terms often result in inaccurate translations. It can also help populating ontologies by providing a candidate term list which can be represented as word vectors that show similarity between terms using word embedding methods.

Even though there are plenty of research in the ATE field, it still remains as an unsolved task because of a few obstacles that hinder the advancements in the field. The most frequently noted issues in the studies are the difficulty in defining what a term is and the lack of datasets generally and especially in languages other than English. Many studies note the disagreement in deciding what a term is since there are many variables like term length, POS patterns and theoretical considerations. Some studies do not even consider extracting multi-word expressions and their methods only extract single-word terms. These problems also lead to disagreements between annotators if the researchers hired multiple domain experts to compile a dataset. The process of annotating manually is also time-consuming and takes a lot of effort which obstructs the collection and creation of large datasets. The scarcity in the datasets also negatively affects the evaluation of the systems. Many studies only compare their methods to older methods using the same dataset and some of them do not even provide an evaluation for their methods. Furthermore, most of the bigger scale studies like shared tasks are done in English and similar structured languages by using English datasets. As of now, there are only two small scaled studies done in the field of ATE in Turkish with none of them being in the legal texts domain and the field remains hugely unresearched.

In this project, an ATE system based on a previous study [1] has been developed. The neural networks used in the system are trained using legal texts compiled from Yargıtay Karar Arama interface using a web-scraping script. The compiled texts are then preprocessed to remove the undesired symbols present in the text. The preprocessed text is then passed to the ITU Natural Language Processing (NLP) Pipeline [2] to apply morphological analysis. Word groups with certain patterns extracted from the analyzed text are the candidates that is used by the system. This process will be explained in further detail in the upcoming sections. The system uses the weakly supervised method proposed in [1] combined with a fine-tuned state-of-the-art word embedding generator, namely BERT [3], to train two different deep learning networks simultaneously. Since there is no dataset provided, the networks start training on a custom labeled dataset that is formed by the terms in the T.C. Adalet Bakanlığı Hukuk Sözlüğü and most frequent 2000 words in the Turkish language provided by the Zemberek NLP [4] study. The co-training algorithm enlarges the starting set by appending a fixed number of best predictions to it. This method also eradicates the need of manually selecting features by using word embeddings. By using the mentioned methodology, this project overcomes the obstacle of having no large, reliable dataset. While there is no good way to evaluate the performance of extraction since there is no annotated dataset, by using the terms in the labeled set as the ground truth, the system can achieve at least 40-45% accuracy with its top 100 predictions. Analyzing the extracted terms also shows that the model can find terms that are not present in the labeled set which implies that the model's real performance is higher. As a result, the system can save huge time for human evaluators that needs to deal with this task and this project can be used as a base for further research in the field.

1.1 Engineering Standards and Multiple Constraints

One of the key principles followed in this project was keeping a separate script for each task and saving all of the intermediate results produced by these scripts. This allowed easily experimenting with different methods throughout the development of the project without going through the same time consuming processes multiple times. In case of internal errors like logic errors or bugs in the code and external errors like power failure, saving the data allowed the continuation of the current process without losing much time. The saved data has also been used for visualization and analysis multiple times which helped with taking decisions and evaluating the project's success. Another crucial point of this project was the lack of a dataset. To remedy this problem, web scraping scripts written from scratch or readily available has been used which saved a considerable amount of time. By the creation of a custom dataset in a semi-automatic manner, different techniques proposed in the literature were made possible to use. The extensive literature review of the project also made choosing a suitable approach for the problem possible.

2 Comparative Literature Survey

2.1 Existing Datasets

There are two manually annotated datasets that have been used in most of the studies done in the field. These are GENIA [5] and ACL RD-TEC 2.0 [6] datasets which are in English. The GENIA corpus is a dataset that includes a compilation of biomedical documents and their annotations according to the GENIA project. The aim of the project was to create a corpus to help the development of information extraction systems in the field of molecular biology. It contains a few types of annotations one of which is term annotations. Around 400k tokens were annotated by two domain experts resulting in around 93k term annotations. The ACL RD-TEC 2.0 is another dataset that has terms annotated by two domain experts from 300 abstracts that is present in the ACL Anthology Reference Corpus. The dataset was created by the purpose of enhancing evaluation of term and entity recognition systems. Around 6.8k terms were annotated from 33k tokens in the field of computational linguistics. Even though ACL RD-TEC 2.0 dataset has less tokens compared to GENIA it has been reported that the dataset offers advantages such as being easier to understand by ATE researchers because of its domain and having more transparent annotation guidelines. Another English dataset is the CRAFT: The Colorado Richly Annotated Full Text Corpus dataset [7]. It provides around 100k term annotations collected from around 560k tokens which were compiled from 67 biomedical journal articles. The annotation guidelines and the dataset are available online to access freely. The ACTER dataset is another manually annotated domain-specific corpora which is provided by a recent study in order to specifically work on term extraction research [8]. ACTER dataset provides annotations made in three languages (English, French and Dutch) and four domains (corruption, dressage, heart failure and wind energy). It offers comparable corpora in all domains provided and has around 50k annotated tokens in each domain. The annotation guidelines for this dataset is also available online and the process is explained in a detailed fashion. Since there is no base guideline for term annotation and every dataset has its own guideline, comparison between different datasets is not a good measure see performance differences while evaluating the model. Because of this reason, studies sometimes include evaluations for multiple datasets and the performance of the methods can vary on different datasets. As an example in Wang et al.[1], the proposed co-training model achieves an F-score of 44.0% on the GENIA N-gram dataset while achieving an F-score of 69.2% on the ACL RD-TEC dataset. As noted in the study, both datasets have 15k test examples and the distribution of positive and negative examples are similar.

As for Turkish, there are no studies that provide such a dataset yet. One study [9] uses a compilation of Turkish and English unannotated texts from cybersecurity domain. The study also provides no evaluation for their methods as they note that there is no reliable source they can evaluate their methods on. Another study [10] uses a Turkish to English medical corpus in the form of a translation memory as their dataset to test the perfomances of different methods without comparing them to a gold standard. This project is also limited by the lack of data since no dataset is provided for the project. Instead the project uses a custom dataset compiled from Yargıtay Karar Arama interface using web-scraping. Hence, it is not possible to properly evaluate or compare the methods proposed by this project either. The lack of Turkish studies in the field shows the importance of having reliable datasets.

2.2 Representing Words and Documents

The task of representing words or documents is one of the most crucial tasks of NLP as the quality and the quantity of the information carried by the representation is essentially what sets the upper limit of the performance on all tasks. Due to its importance, many different ways of generating representations have been proposed over the years. Currently the most useful and popular way of representation is generating vectors as they have many properties that can be used to do various calculations such as addition, subtraction, averaging and measuring distance. Another property that makes vectors a suitable choice is machine learning algorithms that are already available are very intuitive and easy to use with them as they already use vectors.

Choosing vectors as the way to represent words brings us to the task of generating those vectors. As noted in [11], there are two well-known approaches: Vector Space Model (VSM) and statistical language models. Applying analytical methods such as measuring similarity and composition to text data requires a convenient representation. VSM mainly aims to solve this problem. The method suggets representing whole documents as t-dimensional vectors where each dimension corresponds to a term in the document. The values of these dimensions can be any type of number or normalized values according to some metric. Having such vectors allows the calculation of similarity between documents by doing simple operations like dot product or euclidean distance. On the contrary, statistical language models represent words in a way which reckons the probabilistic distribution of them in the text. This allows computations like calculating the likelihood of a word coming after another given word which is useful in tasks like speech recognition [12]. Even though, calculating the probability of combination of words in a language is not feasible the method still performs well enough to achieve satisfactory results. Besides with the employment of neural networks and other methods such as smoothing, it is possible to make the model more robust against unseen phrases. Although two methods differ in some ways, they both depend on the assumption that words in similar contexts have similar meanings. With taking this assumption into account, we can define word embeddings as dense, fixed-length word vectors, that are built using statistical co-occurance data.

Generating meaningful word embeddings is a task on its own and several different ways of accomplishing this task has been proposed in the past. Two of the most prominent methods, namely Skip-gram and Continuous Bag of Words (CBOW) are explained in [13]. Both of these methods use the statistical co-occurence of words in an unsupervised manner to learn the representations of words. CBOW model defines the representation of a word by the summation of the vectors that belong to words that comes both before and after that word without taking their orders into account. Skip-gram also uses a similar idea but instead of guessing the word from the words around it, skip-gram trains a classifier that guesses the surrounding words from the given word. Another very influential model is Global Vectors for Word Representation (GloVe) [14]. This model combines both the VSM and statistical approaches to create a meaningful vector space of word embeddings. This method maps the words into vector in such a way that while similar words have similar vectors, vectors with different meanings will be far from each other. Thus, it also allows finding words that are similar to each other. By having such properties, the method allows the famous "Queen - Woman + Man = King" equation. For a more detailed survey on the topic, reader is referred to [11]. While the system this project is based on ([1]) uses the Skip-gram model, I chose a more modern system that uses state-of-the-art mechanisms like attention and deep neural networks to generate a statistical language model, namely BERT, which will also be explained in detail in the following sections.

2.3 ATE Methodologies

Previous studies on the field can be gathered in three types of methods. Statistical methods which use word frequencies and comparing corpora to each other, linguistic methods which depend on chunking, POS tagging, word embeddings and sometimes neural networks and lastly, hybrid methods which combine the former two methods. Since hybrid methods outperform the other methods, they are preferred more in the studies. Nevertheless, there are a few examples in which pure statistical or linguistic methods are used which are also mostly older studies. As an example for pure statistical methods [15] can be given. In this study, groups of two adjacent words are extracted as candidate terms and then they are ranked according to the TF-IDF statistical measure. Another example is [16] in which groups of words with a chosen length N are first extracted and then ranked according to other empirically chosen measures like frequency and length. As for the pure linguistic methods [17] notes that syntactic data is sufficient to extract terms. Their method firstly eliminates some of the sequences in the text according to their POS tags (for example pronouns and verbs are not considered terms). Then they use a syntactic parser to select the terms from the remaining parts. Lastly, hybrid methods have many different approaches that combine the former two approaches. These methods mostly use linguistic methods as filters in their first stage to get rid of word groups that are unlikely to be terms. Then they apply statistical measures to the selected list of candidates to select the actual terms and rank them between each other. One example for the hybrid methods is the Termolator [18]. As explained in the system description paper, the Termolator uses terminological chunking, distributional ranking and filtering methods to carry out the term extraction operation. It uses two corporas called the foreground and the background corpora to estimate term frequencies and use those metrics to rank the candidate terms. Another hybrid method is explained in [19]. This study combines several methods like TF-IDF, YAKE, RAKE and an algorithm they developed that does clustering on the generated word embeddings. They note that each method is good at capturing certain types of terms and bad at capturing others so they built a system that does voting on the results obtained from each method separately to achieve a better result compared to the results obtained by methods single-handedly. Finally, in [20] we observe a completely different approach compared to other methods. Their method uses a deep learning model called BERT that is known to be highly performant in many NLP tasks. They use the next sentence prediction capability of BERT to extract terms. Positive examples are passed as, for the first sentence, the sentence the term that is located in and for the next sentence to be predicted, the term that is annotated in the dataset. They also generate negative examples in the same format but replace the next sentence to be predicted to a random word from the first sentence that is not a term. There are other studies that also use neural networks and word embeddings to extract terms. In [21] a BiGRU network is used with word embeddings that are concatenated with document embeddings to capture the catchphrases from legal documents. [1] uses a weakly-supervised method that combines the outputs of a CNN and an LSTM which are then used to expand the gold standard. For a more comprehensive investigation of the studies in the field, the reader is referred to [22] which provides great insight about the methodologies and measures that are mostly used and also referred to the more recent [23] which provides a comparison of several methods discussed above.

3 Developed Approach and System Model

The developed system is a pipeline that consists of 3 main components. These components are the filters (morphological analyzer, POS tag parser, pre-processing filters), BERT language model and co-training classifiers. Each of these components and the steps taken to develop them are explained in the following subsections. The whole pipeline will also be summarized in the last subsection.

3.1 Data Collection and Preprocessing

The project aims to extract terms from Turkish legal texts with the help of deep learning methods which brings forth the need of a dataset that contains texts specific to legal domain. Since there is no previous study in the field and no dataset was provided, a custom dataset is created by semi-automatically collecting and processing domain specific text mainly using three tools which are emsal [24], ITU NLP Pipeline [2] and Python scripts written by me.

Emsal is a Python library that allows the semi-automatically downloading of decisions based on year and decision number parameters from the Yargıtay Karar Arama interface.

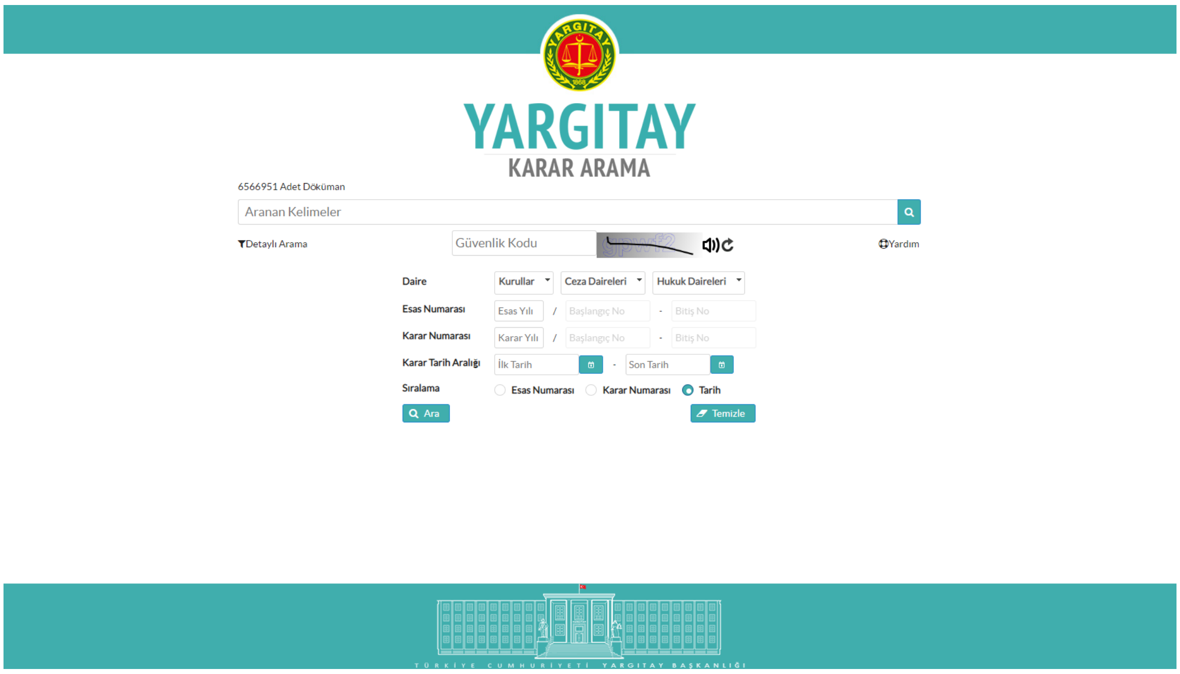


Figure 3.1: Yargıtay Karar Arama Interface

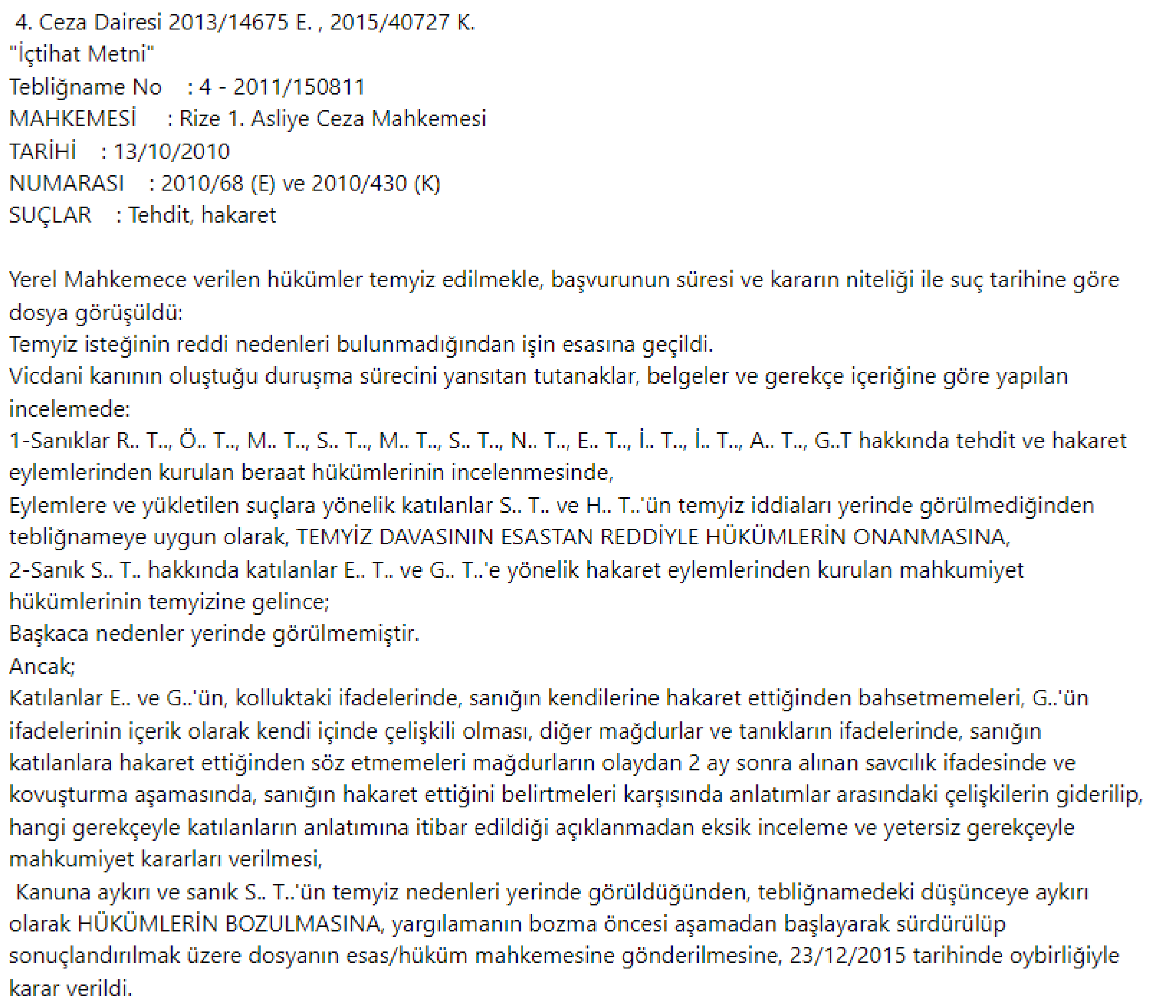
Using the library is a very simple process. After navigating to the Yargıtay Karar Arama interface, the CAPTCHA should be filled correctly. Then, a Python script similar to Figure 3.2 should be run with the desired parameters and downloading process will start.

```
    emsal.get_decisions(
        driver_path="/path/to/chrome/driver",
        year="year",
        init_no=starting_decision_number,
        last_no=ending_decision_number
    )

```

Figure 3.2: Downloading decisions from Yargıtay Karar Arama Interface with emsal

The library uses Selenium [25] and ChromeDriver [26] utilities to navigate through the results of the queries and then simulates user inputs to download and name the files accordingly. Since the Yargıtay Karar Arama interface has a download limit, the script waits 60 seconds after every 30 documents downloaded. Using the script, a total of 7659 documents have been downloaded.



4. Ceza Dairesi 2013/14675 E. , 2015/40727 K.
"İçtihat Metni"
Tebliğname No : 4 - 2011/150811
MAHKEMESİ : Rize 1. Asliye Ceza Mahkemesi
TARİHİ : 13/10/2010
NUMARASI : 2010/68 (E) ve 2010/430 (K)
SUÇLAR : Tehdit, hakaret

Yerel Mahkemece verilen hükümler temyiz edilmekle, başvurunun süresi ve kararın niteliği ile suç tarihine göre dosya görüşüldü:
Temyiz isteğinin reddi nedenleri bulunmadığından işin esasına geçildi.
Vicdani kanının oluştuğu duruşma sürecini yansıtan tutanaklar, belgeler ve gerekçe içeriğine göre yapılan incelemede:
1-Sanıklar R.. T.., Ö.. T.., M.. T.., S.. T.., M.. T.., S.. T.., N.. T.., E.. T.., İ.. T.., İ.. T.., A.. T.., G..T hakkında tehdit ve hakaret eylemlerinden kurulan beraat hükümlerinin incelenmesinde,
Eylemlere ve yükletilen suçlara yönelik katılanlar S.. T.. ve H.. T..'ün temyiz iddiaları yerinde görülmediğinden tebliğnameye uygun olarak, TEMYİZ DAVASININ ESASTAN REDDİYLE HÜKÜMLERİN ONANMASINA,
2-Sanık S.. T.. hakkında katılanlar E.. T.. ve G.. T..'e yönelik hakaret eylemlerinden kurulan mahkumiyet hükümlerinin temyizine gelince;
Başkaca nedenler yerinde görülmemiştir.
Ancak;
Katılanlar E.. ve G..'ün, kolluktaki ifadelerinde, sanığın kendilerine hakaret ettiğinden bahsetmemeleri, G..'ün ifadelerinin içerik olarak kendi içinde çelişkili olması, diğer mağdurlar ve tanıkların ifadelerinde, sanığın katılanlara hakaret ettiğinden söz etmemeleri mağdurların olaydan 2 ay sonra alınan savcılık ifadesinde ve kovuşturma aşamasında, sanığın hakaret ettiğini belirtmeleri karşısında anlatımlar arasındaki çelişkilerin giderilip, hangi gerekçeyle katılanların anlatımına itibar edildiği açıklanmadan eksik inceleme ve yetersiz gerekçeyle mahkumiyet kararları verilmesi,
Kanuna aykırı ve sanık S.. T..'ün temyiz nedenleri yerinde görüldüğünden, tebliğnamedeki düşünceye aykırı olarak HÜKÜMLERİN BOZULMASINA, yargılamanın bozma öncesi aşamadan başlayarak sürdürülüp sonuçlandırılmak üzere dosyanın esas/hüküm mahkemesine gönderilmesine, 23/12/2015 tarihinde oybirliğiyle karar verildi.

Figure 3.3: Example document downloaded from Yargıtay Karar Arama Interface

All of the documents are similarly structured. An example can be observed in Figure 3.3. As shown in Figure 3.3 there are a lot of non-word symbols, initials, punctuation and numbers involved in the documents. These symbols should not be considered when both training the BERT model and extracting terms. Hence, the documents are filtered before the training steps. The initial filtering process is done in 5 consecutive steps. The documents are first loaded into the memory as a single combined string which is formed by concatenating them. As the second step, the documents are made lowercase and tokenized into words. In the third step, punctuation that is present in the tokens are filtered. In the fourth step, the numbers that are in the tokens are filtered. Finally as the last step, the tokens that have a length less 2 than two after the other filtering operations are removed. This last step removes non-word tokens left out like the initials of the suspects which can be seen in the Figure 3.3. After the initial filtering the data is saved as a single text file. After the initial filtering step the produced text is used for three different purposes. Firstly the text file is splitted into two parts as train(70%) and validation(30%) set for further pre-training BERT which is explained in more detail in the next subsection. Secondly, the unsplit text is used for candidate identification which is then used to train the classifiers used in the co-training algorithm. Lastly, terms are extracted using the trained classifiers.

Two different approaches proposed in [1] were used in the process of candidate identification. The first one is the n-gram identifier. This identifier proposes candidates by generating all possible n-grams between two stopwords. This approach is very slow at generating candidates due to the number of possible combinations and did not perform well so it is not included in the final product. The second approach, POS identifier, uses ITU NLP Pipeline to identify candidates. The main idea of this approach is to select word groups that fit to a commonly used POS pattern for term extraction. The chosen pattern is ⟨JJ⟩*⟨NN.*⟩+ which is a number of adjectives followed by a number of nouns. This approach needs a POS tagger since a custom dataset is used and POS tags are not provided. ITU NLP Pipeline offers a morphological analyzer tool which generates the needed POS tags from given strings. Due to the document length limit of the pipeline API, the collected documents were split into 100 equal parts and then passed to the morphological analyzer. The results are then processed with a parser to extract the candidates that fit to the chosen pattern.

The co-training algorithm also requires a small set of labeled data. Again since there was no dataset provided, a custom labeled dataset has been created. This set contains 2000 of the most common non-term words in Turkish provided by Zemberek NLP and around 1500 words from T.C. Adalet Bakanlığı Hukuk sözlüğü which are considered to be terms. As explained in further detail in the following subsections, co-training algorithm uses this labeled dataset as its starting point.

3.2 Word Embedding Generation

The co-training algorithm which will be explained in the following sections needs word embeddings to classify the inputs as terms or not. Thus, the generation of word embeddings is an important part of this project. As shown in many previous studies such as [27], [28] and [29], pre-trained language models are an effective way to increase the performance in various NLP tasks. There are two ways to employ these language models: feature-based and fine-tuning. Both approaches have the goal of learning a general language representation during pre-training but while feature-based approaches such as ELMo[30] use the pre-trained features as an addition to the task-specific features, approaches that use fine-tuning such as BERT introduce very few task-specific features. Instead, they use the existing pre-trained features as a base and improve them by training on a downstream task. This project uses the mentioned BERT to follow a fine-tuning based approach on generating word embeddings.

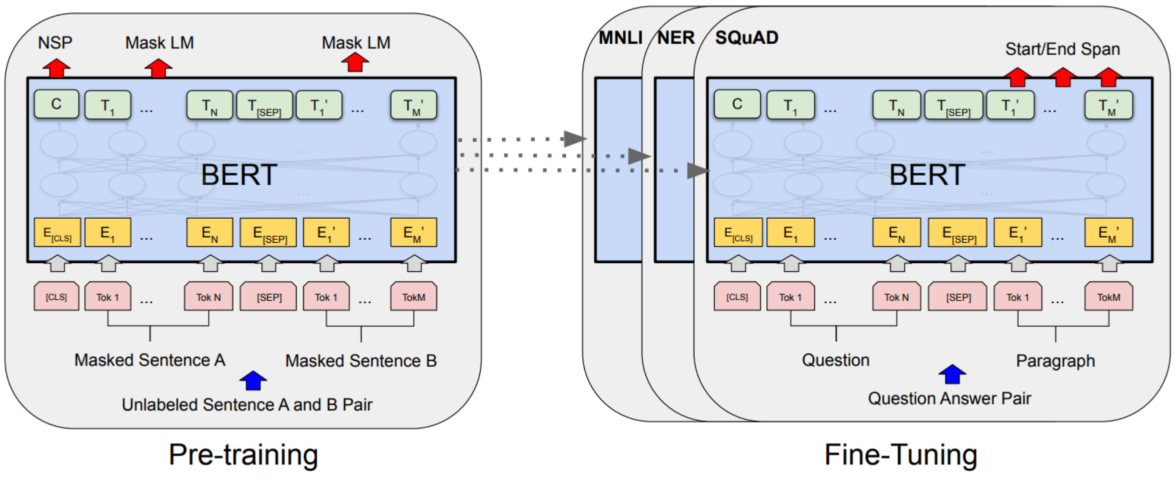


Figure 3.4: BERT pre-training and fine-tuning procedures [3]

BERT is a language representation model that stands for Bidirectional Encoder Representations from Transformers. It is a model that is designed to pre-train bidirectional representations from unlabeled texts using both left and right context. Due to this design it is possible to fine-tune the model on a wide range of tasks such as language inference, question answering and named entity recognition just by adding a single layer on top of it. In order to obtain bidirectional representations, the model uses the Masked Language Model (MLM) as its pre-training objective which is, as noted in the BERT paper, inspired by the Cloze task [31]. Unlike unidirectional language model pre-training, MLM allows the inclusion of both left and right context which can be used to train a bidirectional Transformer. Besides MLM, BERT also uses the next sentence prediction task to train text pairs as another pre-training objective.

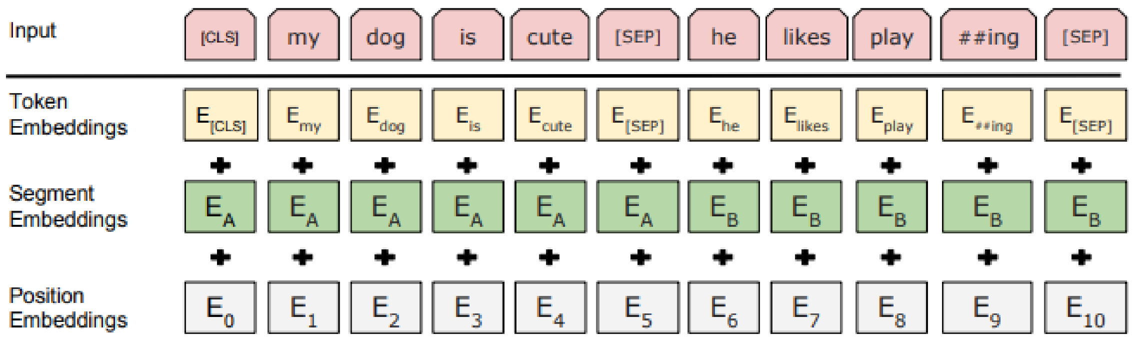


Figure 3.5: BERT input representation [3]

The MLM task randomly masks a percentage of the tokens in the input and BERT predicts these masked tokens. The hidden vectors at the final layer are then passed to a softmax layer over the vocabulary similar to a standard language model. The percentage of input tokens that are masked are 15% in a baseline BERT model. Though MLM allows the training of bidirectional representations, it also comes with a downside. The MLM task creates a mismatch between the pre-training and fine-tuning inputs by introducing a mask token which does not occur in the fine-tuning process. To mitigate this problem, 15% of the tokens are not directly masked. Instead, only 80% of those tokens are masked while 10% of the tokens are replaced with a random token and the remaining 10% of them are not changed at all.

The next sentence prediction (NSP) task aims to train the model to understand the relationship between two sentences which can not be understood directly by language modeling. In this task sentence pairs are generated from the pre-training corpus. These pairs are then fed to the model in a way that 50% of the time the fed next sentence is correct and 50% of the time it is another random sentence from the corpus.

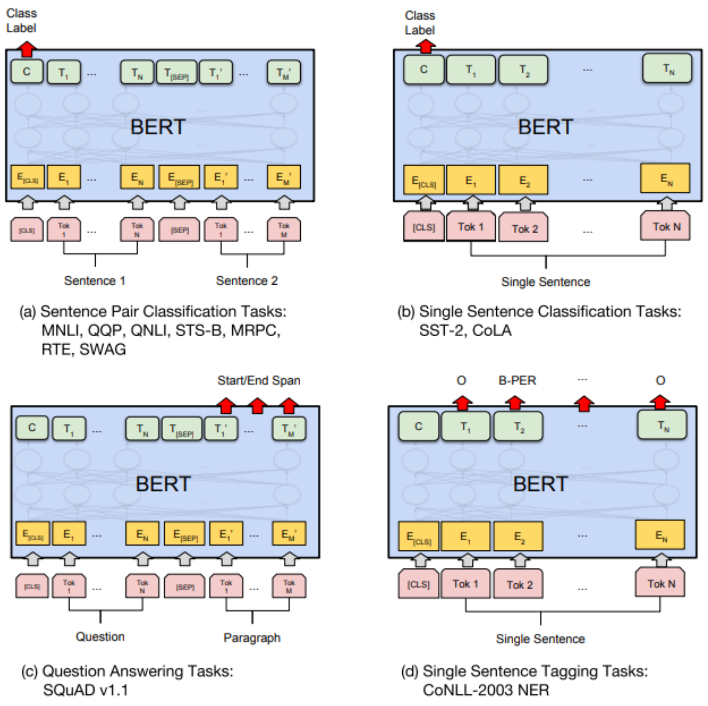

Class Label
BERT
Sentence 1
Sentence 2
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG
Class Label
BERT
Single Sentence
(b) Single Sentence Classification Tasks: SST-2, CoLA
Start/End Span
BERT
Question
Paragraph
(c) Question Answering Tasks: SQuAD v1.1
O
B-PER
O
BERT
Single Sentence
(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Figure 3.6: Fine-tuning BERT on different tasks [3]

After pre-training, the model can be fine-tuned to increase its accuracy on a specific task. As shown in figure 3.6, both pre-training and the fine-tuning uses the same architecture except the output layers. This allows the usage of a pre-trained model on a different task by replacing the output layer with a layer that is suitable to the task and then shortly training the final architecture. In this project, I used a BERT model that is pre-trained on a 35GB Turkish corpus with a vocabulary size of 128k that is publicly available on the Hugging Face API [32]. More information about the pre-training procedure, accuracy of the model and datasets can be found in [33]. Then, I further pre-trained the model on 7659 filtered documents compiled from Yargıtay Karar Arama website to teach the domain specific vocabulary and composition. Further pre-training is done with the MLM approach for 20 epochs with 0.01 weight decay and a batch size of 4. Batch size was limited to 4 due to hardware limitations. Instead of changing the output layer of the model, raw word embeddings were saved to be used by the co-training algorithm. Then, these raw embeddings are classified by the models employed by the co-training algorithm which in a way simulates changing the output layer of the model.

3.3 Co-training Algorithm

Co-training is a weakly supervised learning algorithm which makes training classifiers possible with a dataset that has minimal amount of labeled and a large amount of unlabeled data. The algorithm needs the different views of the same data in which the views are mostly different manually selected features. Different classifiers are trained on these different views of the data. The algorithm can be summarized with the following steps. Co-training starts with training each classifier using the small labeled set for one epoch. Then the classifiers predict a small subset of the unlabeled data. A chosen amount of the most confident predictions among those are added to the labeled dataset to continue training the classifiers. The whole process repeats until the training reaches a chosen iteration count. This project adopts the co-training network architecture that is proposed in [1] that uses neural networks as the classifiers which also eliminates the need of selecting the features by hand.

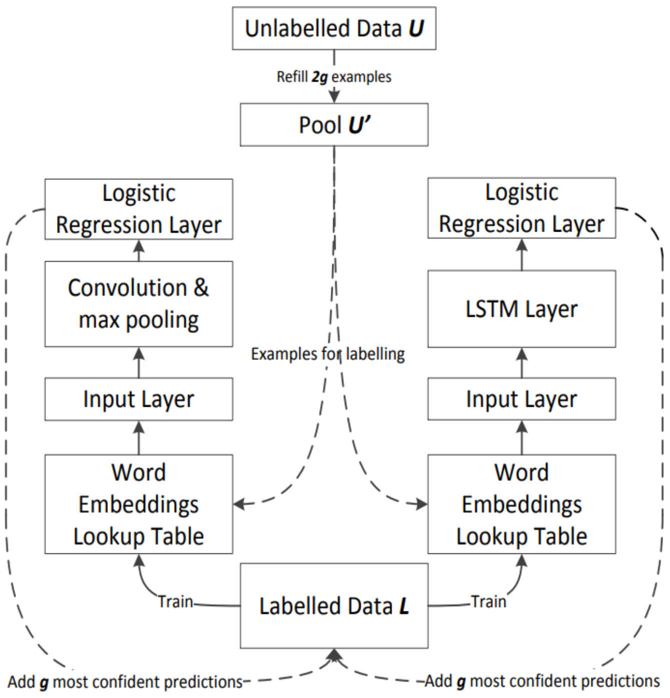

Unlabelled Data ***U***
Refill ***2g*** examples
Pool ***U'***
Logistic Regression Layer
Convolution & max pooling
Input Layer
Word Embeddings Lookup Table
Examples for labelling
Logistic Regression Layer
LSTM Layer
Input Layer
Word Embeddings Lookup Table
Train
Labelled Data ***L***
Train
Add ***g*** most confident predictions
Add ***g*** most confident predictions

Figure 3.7: Co-training architecture overview [1]

As shown in Figure 3.7, the model uses a CNN and an LSTM as the classifiers. The networks use word embeddings generated by BERT to learn the representations of terms. A linear layer connected to a LogSoftmax layer is used as the output layer. This model achieves two different views of the data due to the nature of the chosen networks. While the CNN learns the subgram compositions of the inputs, LSTM learns the order and co-occurance of the same inputs. The architecture of the models can be seen in Figures 3.8 and 3.9.

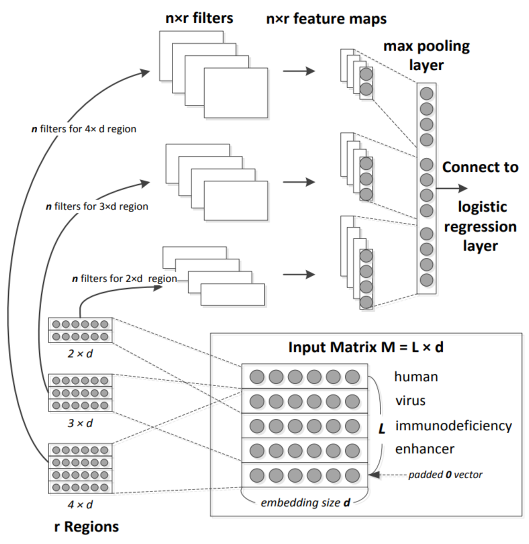


Figure 3.8: CNN model [1]

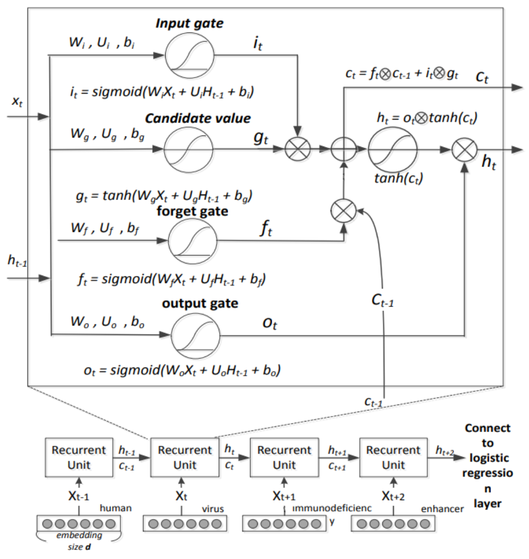


Figure 3.9: LSTM model [1]

The CNN model takes the vertically stacked word embeddings as its input. For a embedding vector of dimension $1 \times d$ the model uses 3 different kernel structures ($2 \times d$, $3 \times d$ and $4 \times d$) to analyze different subgrams of the input. There are 100 of each different kernel structures present in the model. Then 1-Max pooling [34] is applied to the results of each different kernel structure to select the max values for each feature. Then the results obtained from each structure is concatenated to obtain the final feature vector. In this project the dimension of d is selected as 768 since BERT provides 1×768 vectors for its embeddings. LeakyRelu is used as the activation function. The LSTM model directly uses the word embeddings in order. The structure of the LSTM is the same with vanilla LSTM proposed in [35]. Mish [36] is used as the activation function.

A single iteration of the co-training algorithm can be explained more in-depth as the following. For a given unlabeled dataset U and a labeled dataset L, a pool of data U' with size p is selected from the unlabeled set. Firstly each classifier $c \in C$ is trained over the set L. After one epoch, the classifiers make predictions on U'. g most confident predictions from p predictions is selected to enlarge the dataset L. As a result, L now has size $L + 2g$ and U' has size $U' - 2g$. U' is filled with $2g$ random unlabeled data from U. This last operation concludes a single iteration. This operation is iterated k times to complete the training. The pseudo-code can be seen in Figure 3.10.

```
Algorithm 1 Co-training
Input: $L, U, C, p, k, g$
  create $U'$ by randomly choosing $p$ example from $U$
  while $iteration < k$ do
    for $c \in C$ do
      use L to train $C$
    end for
    for $c \in C$ do
      use $c$ to posit label in $U'$
      add most confident $g$ example to $L$
    end for
    refill $U'$ by randomly choosing $2 \times g$ example from $U$
  end while
```

Figure 3.10: Co-training algorithm pseudo-code [1]

3.4 The Whole Pipeline

The whole pipeline can be seen in Figure 3.11.

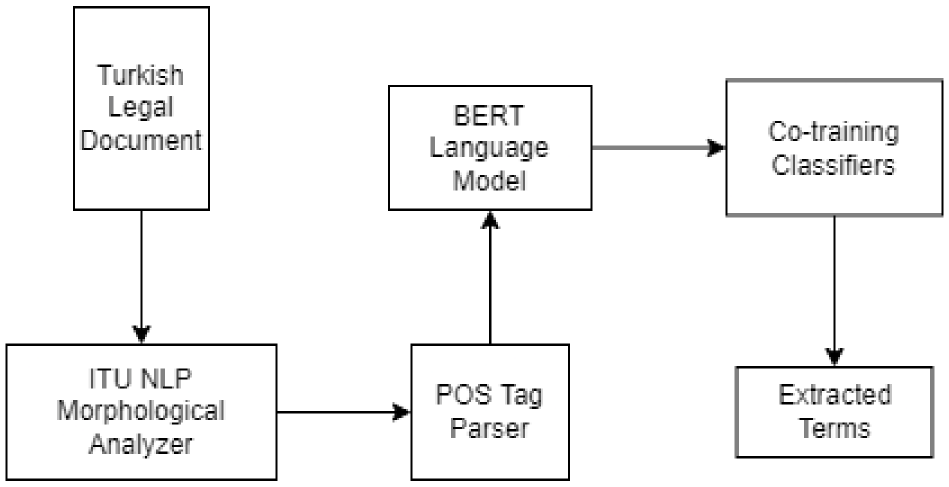


Figure 3.11: The whole pipeline

Briefly the steps in the whole pipeline can be summarized as the following. A Turkish legal document is chosen as the input. The input document is passed through the ITU NLP Morphological Analyzer. The obtained POS tags are then parsed to select the word groups that conform to the pattern of $\langle JJ\rangle*\langle NN.*\rangle+$ which is a number of adjectives followed by a number of nouns. The selected word groups are considered as candidates and they are passed to the BERT language model to obtain their word embeddings. The embeddings are then passed to the classifiers trained by the co-training algorithm. The word groups classified as terms are saved to a file as extracted terms.

4 Experimentation Environment and Experiment Design

A few parameters are used in the co-training algorithm. Different combinations of these parameters have been tested during the development of the pipeline but the final parameters are as the following. The size of the labeled is set to $L = 1000$ and the size of the pool is set to $U' = 1000$. After each prediction $g = 25$ most confident predictions are taken. Number of iterations is set to $k = 100$. Max term length for the classifiers is set to 4 which is selected considering the length of the terms in the labeled dataset. Both classifiers are trained using the AdamW optimizer with a learning rate of $1e-3$. The learning rate decays using StepLR in every 20 steps with gamma selected as 0.1. The loss criterion for both networks is negative-log-likelihood. The batch size is chosen as 128.

The development of the pipeline has started on a Google Colab environment which uses a 12GB GPU but due to training time limitations I had to switch to my local setup. The final development environment uses an Nvidia GTX1050 4GB graphics card and an Intel i5-8300H 2.3GHZ processor on a machine that operates on Windows 10. Visual Studio Code with a virtual environment is used as the IDE. The Python version used in the project is 3.9.9. NLTK is used for tokenizing the documents. PyTorch with CUDA is used as the deep learning framework for the co-training models. Hugging Face is used to obtain the pre-trained BERT model. Finally, Matplotlib and NumPy are used to plot the results and other necessary plots.

5 Comparative Evaluation and Discussion

As stated in the previous subsections, no dataset is provided for the task, therefore all of the algorithms mentioned uses custom datasets created specifically for the project. This makes the comparison of this study to the others infeasible. The lack of a labeled dataset also inhibits properly evaluating the extracted terms since T.C. Adalet Bakanlığı Hukuk Sözlüğü includes only a part of the terms in the legal domain. Nevertheless a baseline for evaluation is provided by comparing the extracted terms to the words in the T.C Adalet Bakanlığı Hukuk Sözlüğü to at least show that the models can extract the words present in the dictionary. Assuming every word in the dictionary is a term, an accuracy plot based on this approach provides the lowest performance the model can achieve since if the words that are not in the dictionary but are extracted by the models are terms, that would only increase the accuracy.

The distribution of the custom labeled dataset is as shown in Figure 5.1.

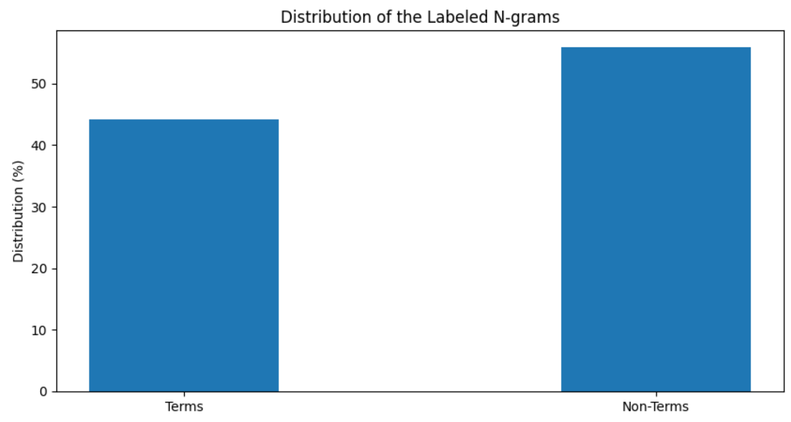


Figure 5.1: Distribution of the labeled dataset

Figure 5.2 shows the distribution of lengths of terms in the labeled set.

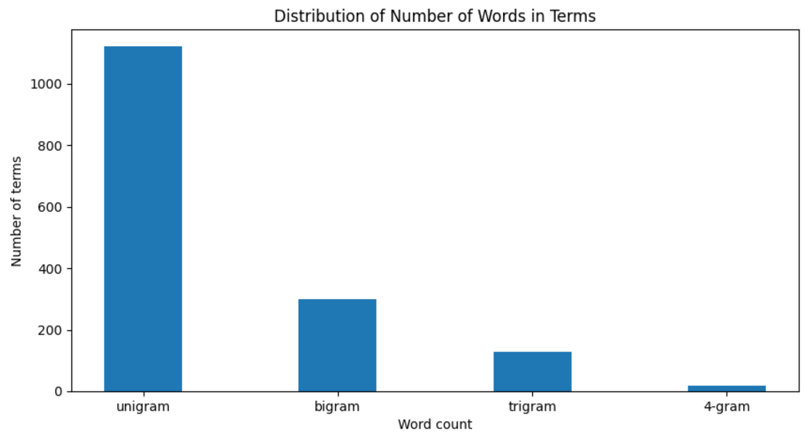


Figure 5.2: Number of words distribution for terms in the labeled set

The dataset heavily leans on unigrams when it comes the length distribution. While more than 70% of the terms are single words, only around 20% of the terms are composed of two words, less than 10% are composed of three words and less than 1% are composed of 1 words. This makes learning the representation of multi-word terms a huge obstacle.

Figures 5.3 and 5.4 show that the classifiers are trained properly.

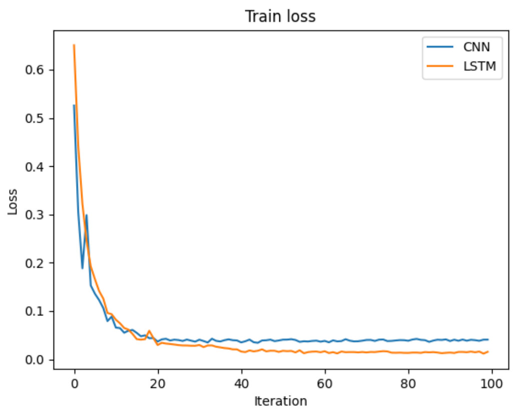


Figure 5.3: Training loss

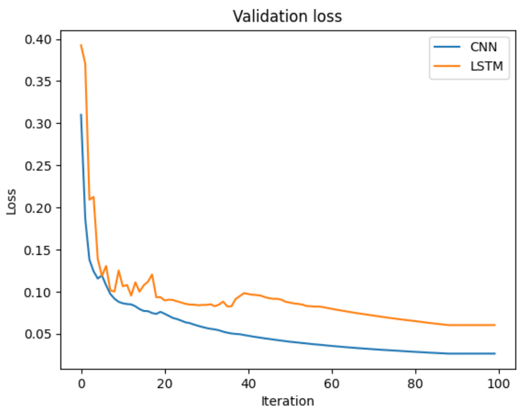
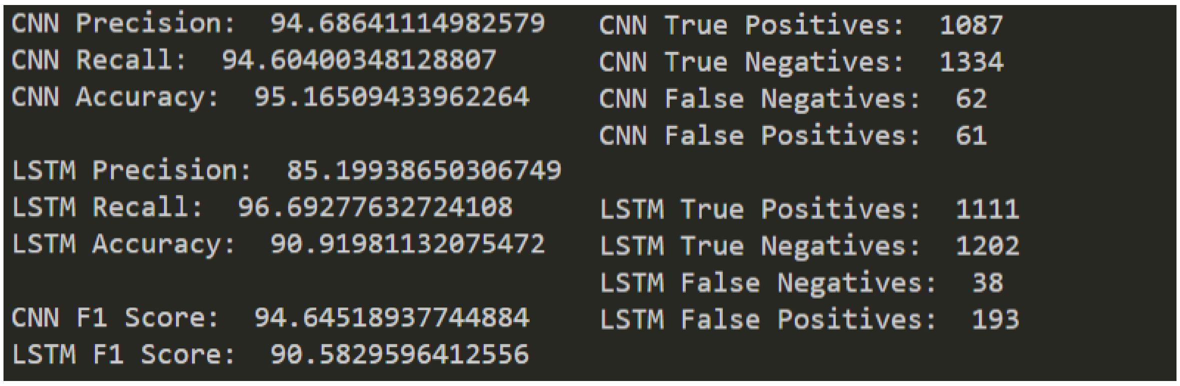


Figure 5.4: Validation loss

Only a part of the labeled set was used in training so the remaining part of this set can be used to show that the models actually learn the representations of the terms.



```
CNN Precision:  94.68641114982579    CNN True Positives:  1087
CNN Recall:  94.60400348128807       CNN True Negatives:  1334
CNN Accuracy:  95.16509433962264     CNN False Negatives:  62
                                     CNN False Positives:  61
LSTM Precision:  85.19938650306749
LSTM Recall:  96.69277632724108      LSTM True Positives:  1111
LSTM Accuracy:  90.91981132075472    LSTM True Negatives:  1202
                                     LSTM False Negatives:  38
CNN F1 Score:  94.64518937744884     LSTM False Positives:  193
LSTM F1 Score:  90.5829596412556
```

Figure 5.5: Testing Results

The test results in Figure 5.5 show that both classifiers can achieve high accuracy on the labeled set which is an indication of correctly learning term representations.

Figure 5.6 shows the accuracy of the models evaluated based solely on the T.C. Adalet Bakanlığı Hukuk Sözlüğü.

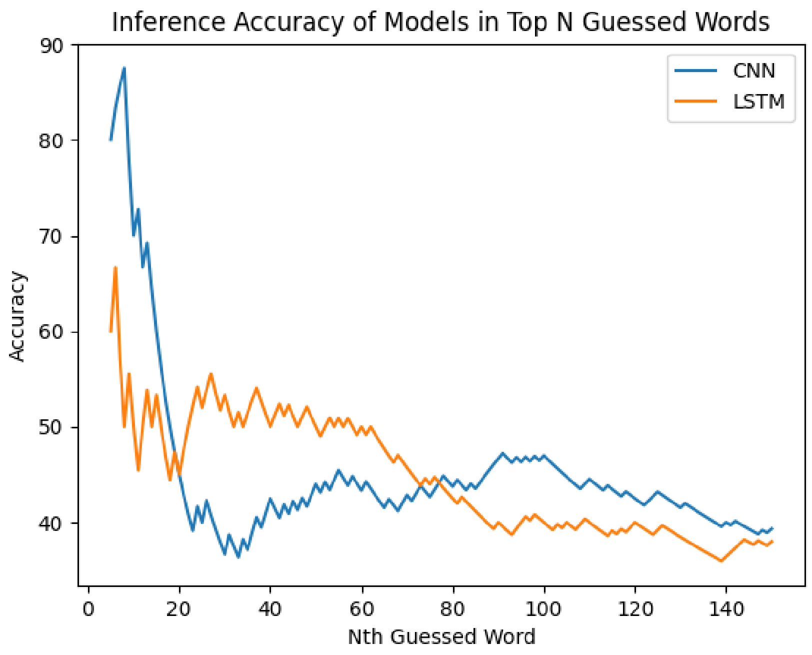


Figure 5.6: Accuracy of models in top N predicted words

The plot shows that the accuracy of the models is around 40% for the top 100 guessed words. As explained in the introduction of this section, this only provides a base for the accuracy of the models. When the output of the models are analyzed it can be seen that they predict words such as ”tapu” which can be considered a term. Even though word groups such as ”Tapuyu misil” and ”Tapu Sicili” exist, ”tapu” does not exist in the dictionary. This shows that the models are able to find terms that do not exist in the labeled dataset. For an accurate evaluation of the models, the extracted terms should be analyzed by a terminologist.

Unfortunately only two out of three of the evaluation criteria mentioned in the interim report can be achieved by this project. 700 candidate terms can be easily extracted by providing more documents. The embeddings of these terms can be also easily obtained and saved. Due to the way BERT model generates its embeddings, it was not possible to get meaningful similarity scores for similar terms.

6 Conclusion and Future Work

The project's main goal of automatically extracting terms from Turkish legal texts is completed. The pipeline developed in this project can be used to extract candidate terms from legal texts to be further evaluated by experts which can be a huge time saver. The extracted terms can be used to improve query results or machine translation. One of the crucial limitations of this project was the dataset. Since there is no Turkish dataset present in the term extraction field, in the future works, a high quality dataset can be compiled by the help of terminologists. The existence of such dataset will allow the usage of different techniques such as state-of-the-art supervised methods which will surely increase the success rate of this task. Having such dataset will also make different methods comparable which is a huge contribution to the field. As explained in this report, in the current state of the field it is not possible to create comparable studies for the Turkish language. The design proposed in this project can be improved by using different word embedding techniques or by using more complex models compared to the CNN and the LSTM. As the final remarks, Turkish ATE still remains as a heavily under-researched field. If more interest and support is shown many more improvements should be expected.

References

[1] R. Wang, W. Liu, and C. McDonald, "Featureless domain-specific term extraction with minimal labelled data," in *Proceedings of the Australasian Language Technology Association Workshop 2016*, Melbourne, Australia, Dec. 2016, pp. 103–112. [Online]. Available: https://aclanthology.org/U16-1011

[2] G. Eryiğit, "ITU Turkish NLP web service," in *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. Gothenburg, Sweden: Association for Computational Linguistics, April 2014.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018. [Online]. Available: https://arxiv.org/abs/1810.04805

[4] A. A. MD Akın, "An open source natural language processing library for turkic languages: Zemberek," vol. 431, 2007, pp. 38–44.

[5] J.-D. Kim, T. Ohta, Y. Tateisi, and J. Tsujii, "GENIA corpus—a semantically annotated corpus for bio-textmining," vol. 19, no. 1, 07 2003, pp. 180–182. [Online]. Available: https://doi.org/10.1093/bioinformatics/btg1023

[6] B. QasemiZadeh and A.-K. Schumann, "The ACL RD-TEC 2.0: A language resource for evaluating term extraction and entity recognition methods," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), May 2016, pp. 1862–1868. [Online]. Available: https://aclanthology.org/L16-1294

[7] C. K. L. A. W. C. J. H. R. C. C. J. F. C. M. Y. E. M. X. N. B. J. W. B. M. P. M. H. L. Verspoor, K.*, "A corpus of full-text journal articles is a robust evaluation tool for revealing differences in performance of biomedical natural language processing tools." vol. 13, no. 1, Aug. 2021, p. 207.

[8] A. Rigouts Terryn, "ACTER (annotated corpora for term extraction research) v1.3," 2019, eurac Research CLARIN Centre. [Online]. Available: http://hdl.handle.net/20.500.12124/24

[9] İrfan Aygün and M. Kaya, "Automatic term extraction on turkish scientific texts," *2020 International Conference on Decision Aid Sciences and Application (DASA)*, pp. 1037–1040, 2020.

[10] G. Dogru, "Automatic term extraction from turkish to english medical corpus," *Computational and Corpus-based Phraseology*, vol. 157, 2019.

[11] F. Almeida and G. Xexéo, "Word embeddings: A survey," *CoRR*, vol. abs/1901.09069, 2019. [Online]. Available: http://arxiv.org/abs/1901.09069

[12] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Transactions on Pattern Analysis and*

Machine Intelligence, vol. PAMI-5, no. 2, pp. 179–190, Mar. 1983. [Online]. Available: https://doi.org/10.1109/tpami.1983.4767370

[13] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013. [Online]. Available: https://arxiv.org/abs/1301.3781

[14] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[15] G. Salton, C. S. Yang, and C. T. Yu, "A theory of term importance in automatic text analysis," *Journal of the American Society for Information Science*, vol. 26, no. 1, pp. 33–44, 1975. [Online]. Available: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.4630260106

[16] L. P. Jones, E. W. G. Jr., and S. Radhakrishnan, "INDEX: the statistical basis for an automatic conceptual phrase-indexing system," *J. Am. Soc. Inf. Sci.*, vol. 41, no. 2, pp. 87–97, 1990. [Online]. Available: https://doi.org/10.1002/(SICI)1097-4571(199003)41:287::AID-ASI2\protect\protect\leavevmode@ifvmode\kern+.2222em\relax3.0.CO;2-8

[17] D. Bourigault, "Surface grammatical analysis for the extraction of terminological noun phrases," in *COLING 1992 Volume 3: The 14th International Conference on Computational Linguistics*, 1992. [Online]. Available: https://aclanthology.org/C92-3150

[18] P. Mayr, C. Zhang, A. Meyers, Y. He, Z. Glass, J. Ortega, S. Liao, A. Grieve-Smith, R. Grishman, and O. Babko-Malaya, "The termolator: Terminology recognition based on chunking, statistical and search-based scores," *Frontiers in Research Metrics and Analytics*, vol. 3, 06 2018.

[19] V. Pais and R. Ion, "TermEval 2020: RACAI's automatic term extraction system," in *Proceedings of the 6th International Workshop on Computational Terminology*. Marseille, France: European Language Resources Association, May 2020, pp. 101–105. [Online]. Available: https://aclanthology.org/2020.computerm-1.14

[20] A. Hazem, M. Bouhandi, F. Boudin, and B. Daille, "TermEval 2020: TALN-LS2N system for automatic term extraction," in *Proceedings of the 6th International Workshop on Computational Terminology*. Marseille, France: European Language Resources Association, May 2020, pp. 95–100. [Online]. Available: https://aclanthology.org/2020.computerm-1.13

[21] S. G. S. M. Arpan Mandal, Kripabandhu Ghosh, "A sequence labeling model for catchphrase identification from legal case documents," *Artificial Intelligence and Law*, pp. 1–34, 2021.

[22] P. Maria Teresa, M. Pennacchiotti, and F. M. Zanzotto, *Terminology Extraction: An Analysis of Linguistic and Statistical Approaches*, 06 2006, vol. 185, pp. 255–279.

[23] A. Rigouts Terryn, V. Hoste, P. Drouin, and E. Lefever, "TermEval 2020: Shared task on automatic term extraction using the annotated corpora for term extraction research (ACTER) dataset," in *Proceedings of the 6th International Workshop on Computational Terminology*. Marseille, France: European Language Resources Association, May 2020, pp. 85–94. [Online]. Available: https://aclanthology.org/2020.computerm-1.12

[24] D. Arslan, "emsal," https://github.com/Bilirkisi/emsal, 2021.

[25] "Selenium," https://www.selenium.dev/, accessed: 2022.

[26] "Chromedriver - webdriver for chrome," https://chromedriver.chromium.org/, accessed: 2022.

[27] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., vol. 28. Curran Associates, Inc., 2015. [Online]. Available: https://proceedings.neurips.cc/paper/2015/file/7137debd45ae4d0ab9aa953017286b20-Paper.pdf

[28] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 2227–2237. [Online]. Available: https://aclanthology.org/N18-1202

[29] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 328–339. [Online]. Available: https://aclanthology.org/P18-1031

[30] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018. [Online]. Available: https://arxiv.org/abs/1802.05365

[31] W. L. Taylor, ""cloze procedure": A new tool for measuring readability," *Journalism Quarterly*, vol. 30, no. 4, pp. 415–433, Sep. 1953. [Online]. Available: https://doi.org/10.1177/107769905303000401

[32] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, "Huggingface's transformers: State-of-the-art natural language processing," *CoRR*, vol. abs/1910.03771, 2019. [Online]. Available: http://arxiv.org/abs/1910.03771

[33] S. Schweter, "Berturk - bert models for turkish," Apr. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3770924

[34] Y. Zhang and B. C. Wallace, "A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification," *CoRR*, vol. abs/1510.03820, 2015. [Online]. Available: http://arxiv.org/abs/1510.03820

[35] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *CoRR*, vol. abs/1503.04069, 2015. [Online]. Available: http://arxiv.org/abs/1503.04069

[36] D. Misra, "Mish: A self regularized non-monotonic neural activation function," *CoRR*, vol. abs/1908.08681, 2019. [Online]. Available: http://arxiv.org/abs/1908.08681