RETRIEVING TURKISH PRIOR LEGAL CASES WITH DEEP LEARNING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR
THE DEGREE OF
MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

By
Ceyhun Emre Öztürk
June 2023

RETRIEVING TURKISH PRIOR LEGAL CASES WITH DEEP LEARNING

By Ceyhun Emre Öztürk

June 2023

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Aykut Koç(Advisor)

Tolga Çukur

Murat Can Ganiz

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

RETRIEVING TURKISH PRIOR LEGAL CASES WITH DEEP LEARNING

Ceyhun Emre Öztürk
M.S. in Electrical and Electronics Engineering
Advisor: Aykut Koç
June 2023

This study utilizes deep learning models to retrieve prior legal cases in the Court of Cassation in Turkey. Given the vast legal databases that legal professionals need to navigate and the ability of computers to handle large amounts of text quickly, information retrieval algorithms prove beneficial for legal practitioners. In this thesis, we introduce our legal recurrent neural network (RNN) models and the BERTurk-Legal model. We also introduce dense word embeddings for the Turkish legal domain. Moreover, we employ RNN autoencoders, Legal RNN autoencoders, combinations of RNN autoencoders with BM25 algorithms, and BERTurk-Legal to retrieve prior legal cases. We obtain the best results with the BERTurk-Legal model.

Keywords: Natural language processing, law, deep learning, information retrieval, NLP in law, AI in law, prior legal case retrieval.

ÖZET

DERİN ÖĞRENME İLE TÜRKÇE EMSAL KARAR BULMA

Ceyhun Emre Öztürk
Elektrik Elektronik Mühendisliği, Yüksek Lisans
Tez Danışmanı: Aykut Koç
Haziran 2023

Bu çalışma Yargıtay'da emsal karar bulmak için derin öğrenme modelleri kullanmaktadır. Hukuk alanında çalışanların incelemesi gereken geniş çaplı hukuk veri tabanları ve bilgisayarların büyük miktarda metni kısa zamanda işleme kapasiteleri sebebiyle bulgetir algoritmaları hukuk alanında çalışanlar için faydalıdır. Bu tezde hukuksal yinelemeli sinir ağları (RNN) ve BERTurk-Legal modeli tanıtılmaktadır. Ayrıca Türkçe hukuk alanına özel yoğun kelime temsilleri tanıtılmaktadır. Buna ek olarak emsal karar yöntemi olarak RNN otokodlayıcılar, hukuksal RNN otokodlayıcılar, RNN otokodlayıcıların BM25 ile birleşimleri ve BERTurk-Legal modeli kullanılmıştır. En iyi sonuçlar BERTurk-Legal modeli ile elde edilmiştir.

Anahtar sözcükler: Doğal dil işleme, hukuk, derin öğrenme, bulgetir, hukukta doğal dil işleme, hukukta yapay zeka, emsal karar bulma.

Acknowledgement

Firstly, I would like to thank my advisor Asst. Prof. Aykut Koç. During my M.S. studies, he helped me improve myself in terms of technical and soft skills that are required for working as a researcher. I learned most of my academic writing skills from him. I am going to make use of his advice and feedback throughout my whole career.

I would also like to thank the rest of my thesis committee: Assoc. Prof. Tolga Çukur and Assoc. Prof. Murat Can Ganiz, for their time and valuable feedback.

Also, I would like to thank my current and former colleagues at ASELSAN Inc., especially Dr. Veysel Yücesoy, who always helped me, Dr. Ömer Köksal, Eyüp Halit Yılmaz, Furkan Şahinuç, Dr. Çağrı Toraman, Ümitcan Şahin, and Oğuzhan Özçelik, who participated in valuable discussions with me in the field of NLP, and Emre Altuntaş and Talha İnce for their mentorship.

Moreover, I would like to thank my close friends, including Murat Bilgiş, Rıza Kürşat Özdemir, Süleyman Taylan Topaloğlu, Can Soygür, Onur Özder, Doğa Gürgünoğlu and Ömer Musa Battal. They always provided physical and mental support for me during difficult times. I also thank all my friends from my undergraduate and graduate time at Bilkent, and my lecturers from all the schools I have attended.

I owe to my family and relatives for my achievements. Throughout my life, I got their support in all aspects of life, including mental, physical, and financial support, help with household chores, and many other things. I am who I am today, thanks to them.

This work is supported by TÜBİTAK within the scope of 2210/A scholarship, and in part by TÜBİTAK 1001 grant (120E346).

Contents

List of Figures

List of Tables

List of Publications

This thesis includes content from following publications:

1. A. C. Aras, C. E. Öztürk and A. Koç, "Feedforward Neural Network Based Case Prediction in Turkish Higher Courts," *2022 30th Signal Processing and Communications Applications Conference (SIU)*, Safranbolu, Turkey, 2022, pp. 1-4.

2. C. E. Öztürk, Ö. Köksal, Ş. B. Özçelik and A. Koç, "Prior Case Retrieval for the Court of Cassation of Turkey," *2022 16th International Conference on Signal Image Technology & Internet Based Systems (SITIS)*, Dijon, France, 2022, pp. 539-544.

List of Abbreviations

NLP	natural language processing
LJP	legal judgment prediction
SOTA	state-of-the-art
TF-IDF	term frequency-inverse document frequency
CBOW	continuous bag-of-words
RNN	recurrent neural network
GRU	gated recurrent unit
LSTM	long short-term memory
BERT	bidirectional encoder representations from transformers

Chapter 1

Introduction

Law is a field that consistently changes to satisfy the needs of societies. Law professionals have an increasing burden due to the changes in the field of law. On the other hand, computers can process large amounts of legal texts quickly, increase the accuracy of legal judgments and reduce bias in the legal domain. Therefore, legal natural language processing (NLP) is a popular and precious research topic. Legal NLP focuses on reducing the excessive workload of experts in the legal domain. Prior legal case retrieval [1, 2], legal judgment prediction (LJP) [3, 4, 5], court opinion generation [6], information extraction [7], named entity recognition (NER) [8], and document summarization [9] are some of the popular tasks in legal NLP.

In modern NLP applications, texts from natural languages are mapped into numbers to be processed with statistical formulas or machine learning (ML) algorithms. In the early years of NLP research, words were represented by sparse vectors generated by the term frequency-inverse document frequency (TF-IDF) method [10]. In the 2010s, point word embedding models that represent words with dense word vectors were used in many low-level (intrinsic) and high-level (extrinsic) NLP tasks [11, 12, 13, 14, 15, 16, 17]. [18] created Law2Vec, dense word vectors for the English legal domain. They were motivated by the fact that legal texts include many words rarely used in daily life. Inspired by Law2Vec, we

propose LawTurk2Vec, dense word vectors for the Turkish legal domain. Recurrent neural networks (RNNs) are commonly used in problems where sequential data are processed. These models were widely used in recent legal NLP papers [3, 5, 7]. In this thesis, we use RNN variants to create document embeddings from word embeddings of the legal case decision texts. Then, we compare the similarity of the document embeddings to retrieve prior legal cases. The utilized RNN variants are gated recurrent unit (GRU) and long short-term memory (LSTM). In addition to these RNN variants, we propose RNN models dedicated to the legal domain. These models are named Legal GRU and Legal LSTM. They utilize both word embeddings created for general use and those created for the legal domain, i.e., LawTurk2Vec.

Traditional word embeddings do not take the context of the words into account. However, words can have different meanings depending on the words that surround them. As a result, [19, 20, 21, 22] proposed contextual language models to represent the meanings of the words better. These language models evaluated the representations of the words based on their contexts. They achieved state-of-the-art (SOTA) performance on numerous high-level NLP applications, such as text classification and question answering. Therefore, we introduce BERTurk-Legal as a contextual language model for the Turkish legal domain.

The contributions of this thesis are the introduction of the prior legal case retrieval task for the Turkish Court of Cassation, the LawTurk2Vec word embeddings, Legal RNN models, and the BERTurk-Legal model. This thesis is focused on two published conference articles [4] and [2], which are the outputs of the Master of Science studies. [4] introduces LawTurk2Vec, and [3] introduces the prior legal case retrieval task for the Turkish Court of Cassation.

We organize the sections of the thesis as follows. Chapter 2 mentions the works in legal NLP. Chapter 3 reviews the language models and word embeddings. We describe the details of our data and retrieval task in Chapter 4. We describe the usage of RNN autoencoders to retrieve prior legal cases and provide the results for the models in Chapter 5. We also introduce Legal RNNs and LawTurk2Vec and

their use in RNN autoencoders in this chapter. Then, we describe the BERTurk-Legal model, which we created to retrieve prior legal cases, in Chapter 6 and provide the results for the model. Finally, the thesis concludes in Chapter 7 by giving a final discussion of all results and possible future work.

Chapter 2

Legal NLP

In this chapter, we review the works utilizing NLP methods to solve problems in the legal domain. The first NLP methods utilized in the legal domain are case-based reasoning methods [23, 24, 25, 26] and rule-based algorithms [27, 28]. [29] reviews the first works in legal NLP. In 2000s, traditional ML algorithms, such as decision tree (DT), random forest (RF), and support vector machine (SVMs), became popular in legal NLP [30, 31, 32, 33]. ML algorithms were utilized in the legal systems of many countries, including China [34, 35], the Philippines [36], the UK [37], and Switzerland [38]. Also, legal systems of international organizations such as ECtHR [39, 40, 41, 42, 43] were investigated by NLP researchers.

As the graphical processing units (GPUs) with more computing capabilities emerged, many works in legal NLP [5, 44] utilized deep learning (DL) algorithms, such as feedforward neural network (FFNN), LSTM, and convolutional neural network (CNN). [45] reviews the works in legal NLP utilizing DL algorithms. After [46] introduced their contextual language model architecture named transformer, which is an advanced DL algorithm focusing on attention layers, many other transformer-based language models, such as Bidirectional Encoder Representations from Transformers (BERT) [47], GPT-3 [48], and T5 [49], were proposed. These models provided SOTA performance for many NLP tasks. [1, 50, 51] proposed SOTA BERT-based methods for the legal domain. [52, 53] introduced

standardized benchmarks for legal NLP. Transformer-based models provide SOTA results on these benchmarks.

NLP in the Turkish legal domain is an emerging research field. [5] is the first work in the literature that introduces legal NLP in the Turkish legal system. They introduce an LJP task for the Turkish legal system. They use traditional machine learning methods and neural networks to predict the decisions of the Supreme Courts of Turkey. The learning methods take the decision texts of the courts with the sentences about the last decisions removed as input. The investigated labels are binary, i.e., "rejection" and "admission". The investigated cases in [5] are the Constitutional Court, the Civil Court of Appeal, the Criminal Court of Appeal, the Administrative Court of Appeal, and the Court of Appeal on Taxation [5]. [3, 4, 54] also worked on the LJP task for Turkish legal courts. For the LJP task, [3] used recurrent neural networks (RNNs), [4] used an FFNN, and [54] utilized traditional machine learning models, RNNs, and BERT-based models. [8] proposed a model to detect the named entities in Turkish legal cases. The model consists of an LSTM and a CRF conditional random field (CRF) model connected in series. [55] detected gender bias in Turkish legal texts and adopted the methods proposed in [56] to reduce the effects of gender bias on dense word vectors.

Competition on Legal Information Extraction/Entailment (COLIEE) is a popular legal NLP competition, which includes a prior legal case retrieval task [57]. [58] used a convolutional neural network (CNN)-based model for the retrieval task. The model has a convolutional layer that processes n-gram phrases from sentences. After this layer, there is a sentence level max-pooling layer and a document level max-pooling layer. After that, the output of the document level max-pooling layer is fed to an FFNN. Then, the model outputs are substituted into a deterministic formula to create document vectors. The dot product between the document vector of the query and each document is calculated to order the documents. They won the COLIEE 2019 competition with 57.64% micro-F1, 60.00% micro-precision, and 55.45% micro-recall scores [58]. In the prior legal case retrieval task of COLIEE 2021, queries are decision texts. Therefore, the lengths of the queries are longer than typical queries of Web page retrieval tasks.

The best-performing team of COLIEE 2021, TLIR [1], used LMIR, a traditional IR algorithm. TLIR converted the texts to lowercase. They removed stop words and punctuation. Then, they applied stemming to the words. LMIR calculates the probability of generating the correct query from the candidate document by multiplying the probability of generating each token of the query given the candidate document with each other. With this method, they achieved 19.17% micro-F1, 15.33% micro-precision, and 25.56% micro-recall scores. The winning team also submitted the results for another method that includes the LMIR algorithm and a BERT-based model. This method ranks the documents based on LMIR and then re-ranks the results with a BERT-based model. However, the method's performance was worse than LMIR [1]. In [59], three different methods were used. All methods divide texts into paragraphs. The final retrieval scores of the documents are found by summing up the retrieval scores of the paragraphs. The best-performing method was Okapi BM25, with 11.13% micro-F1, 7.77% micro-precision, and 19.59% micro-recall scores. The second method used two BERT-based siamese encoders, and the other method used the combination of the Okapi BM25 algorithm and the encoders of the second method [59].

[60] developed a prior legal case retrieval system for the Indian Supreme Court. The system tags sentences containing evidence and testimony with a bidirectional LSTM model. The system evaluates the retrieval scores of the documents by processing the tagged sentences with a transformer-based model. [61] processed the decision texts with the LEGAL-BERT model proposed by [50]. [61] divided decision texts into parts to fit the texts into the LEGAL-BERT model. The document vectors of the decision texts were found by summing the document vectors generated by LEGAL-BERT for their parts. Prior legal cases were then determined according to the similarity of the document vectors.

Chapter 3

Language Models and Word Embeddings

3.1 Traditional Word Embeddings

NLP methods require texts to be mapped into mathematical forms, such as document vectors, sequences of word vectors, etc. As a result of this necessity, language models, such as TF-IDF [10], were developed. [62] proposed a neural network language model which consists of a feedforward neural network and produces dense word vectors. [63, 11] proposed two neural network language models, continuous bag-of-words (CBOW) and skip-gram, which produce word2vec embeddings. These models require fewer sources for training than the model of [62] since they exclude the hidden layers before the output layer of the feedforward neural network architecture of [62]. With the introduction of word2vec embeddings and the rising popularity of neural networks in research, word embeddings were used in many NLP tasks as an input to the neural networks [64, 65, 5].

The CBOW model extracts a window surrounding a target word and tries to predict the target word based on the words of the window, i.e., the context words. Vectors of the context words are mapped into a context vector by averaging the

vectors of the context words. Hence, the order of the context words is not taken into consideration. The Skip-gram model follows a similar approach. On the other hand, it takes a target word from the text and predicts the words in the context window which surrounds the target word. The objective of the Skip-gram model is to maximize the following expression:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-c\leq j\leq c\\ j\neq 0}}\log p(w_{t+j}|w_t), \tag{3.1}$$

where T is the number of words in the training corpus, w_t is the target word, and $2c$ is the number of context words [11]. The probability $p(w_{t+j}|w_t)$ is defined as follows:

$$p(w_o|w_i) = \frac{exp({v'_{w_o}}^T v_{w_i})}{\sum_{m=1}^{W} exp({v'_{w_m}}^T v_{w_i})}, \tag{3.2}$$

where v_w and v'_w are the input and output vectors of word w, and W is the cardinality of the vocabulary [11]. By training the Skip-gram model, these word vectors capture the word similarity and analogy relations. By applying simple linear operations on these vectors, questions related to country-capital, man-woman, and some grammatical structures can be solved. For example, when the vector $x = v(\text{"}England\text{"}) - v(\text{"}London\text{"}) + v(\text{"}Paris\text{"})$ is calculated, the resulting vector x is closest to the vector $v(\text{"}France\text{"})$ in terms of cosine similarity as expected.

The GloVe model was proposed in 2014 [12]. The model trains the word embeddings by using the co-occurrence information of the words of the corpus of the model. The idea behind this training procedure is the assumption that similar words of a corpus are close to each other, and thus these words have high co-occurrence counts. [12] conducted a qualitative analysis to show that this assumption holds. The loss function of the GloVe model is provided below:

$$J = \sum_{i,j=1}^{V} f(X_{i,j})(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij})^2, \tag{3.3}$$

where V is the cardinality of the vocabulary, i and j are the indices of the words, w_i and $\tilde{w}_i$ are the word vector and context vector of word i, $X_{i,j}$ is the co-occurrence value for the words i and j, b's are bias terms, and $f(.)$ is a weighting function. $f(.)$ is used to reduce the weight of rare and frequent co-occurrences on the loss value. Its definition is provided below:

$$f(x) = \begin{cases} (x/x_{max})^a, & x < x_{max} \\ 1, & \text{otherwise} \end{cases} \tag{3.4}$$

where x_{max} and a are the hyperparameters of the GloVe model [12].

3.2 Transformer-based Language Models

Since traditional word embedding models, such as Word2Vec [63, 11] and Glove [12], map words to dense vectors, they require fewer computing resources and perform better than primitive language models, which encode information with sparse vectors. However, they do not take the context of the words into account. Although the neural network models that can process input word vectors as sequences, such as recurrent neural networks (RNN) and convolutional neural networks (CNN) [66], take the order of the word vectors into account, the performance of these models is insufficient for the complex tasks, such as text classification and question answering. [67, 68] proposed incorporating attention mechanisms into the neural networks to increase their performance on NLP applications. Before the introduction of attention mechanisms, RNNs encoder-decoder models were widely used for machine translation. The RNN encoder model encoded a text from the source language to create a text vector. Then, the text vector was decoded by the RNN decoder model to generate the text for the target language. However, this procedure tucks all input text information into the text vector, which is too small to keep the information of the encoder model. [67]

added an attention layer to the RNN encoder-decoder architecture to carry the context information to the decoder model with less loss of information. Inspired by this attention mechanism, [19] created the transformer architecture in 2017. They removed the RNN models from the architecture of [67] and processed texts with the transformer model, which consists of only attention layers.

After the introduction of the transformer architecture, many works [21, 22, 49] utilized the architecture to create transformer-based language models, such as BERT [21], GPT-3 [22], and T5 [49]. The transformer-based models are robust since they are easily adaptable for transfer learning approaches. They are typically pre-trained, and the pre-trained model can be fine-tuned on specific tasks for use on these tasks. Pre-training is an unsupervised learning procedure where a transformer-based model is trained with a general task, such as masked language modeling and next sentence prediction, using a large corpus. On the other hand, fine-tuning is a supervised learning procedure where a transformer-based model is trained with a specific high-level task, such as text classification and question answering. Currently, transformer-based models are the best-performing models in many high-level NLP tasks, including text classification and question answering.

Chapter 4

Data Preparation and the Retrieval Task

We introduce the Turkish Legal Corpus to train our retrieval models in an unsupervised manner. It consists of unlabeled decision texts from the Turkish Constitutional Court, the Turkish Court of Cassation, and four regional courts of Turkey, namely the Civil Court of Appeal, the Criminal Court of Appeal, the Administrative Court of Appeal, and the Court of Appeal on Taxation. The above courts are the Higher Courts of the Republic of Turkey, and all court verdicts used to compile the corpus are publicly available. The corpus consists of $2,291,258$ word types. The number of cases from each court is shown in Table 4.1. All training samples are unlabeled since tagging decision texts takes a long time.

We also created a retrieval dataset to test our models. Ş. Barış Özçelik, who is an associate professor at the Faculty of Law of Bilkent University, selected 26 sentences that are common topics for the cases of the Court of Cassation. These sentences and their translations are provided in Tables 4.2 and 4.3. Cases of the Turkish Court of Cassation are publicly available on the official website of the Court of Cassation[1]. We searched for the sentences given in Tables 4.2 and 4.3 on

[1]https://karararama.yargitay.gov.tr

Table 4.1: The number of cases from each court in the Turkish Legal Corpus.

Court Name	**Number of Cases**
Court of Cassation	332,662
Constitutional Court	6,485
Civil Court of Appeal	47,796
Criminal Court of Appeal	9,241
Administrative Court of Appeal	20,948
Court of Appeal on Taxation	8,870

the website of the Court of Cassation and downloaded the top 10 results for each sentence. A screenshot of the website is shown in Figure 4.1. The downloaded cases were not included in our training dataset. Two of the sentences had less than 10 related legal cases (One of the sentences had 9 related legal cases and the other had 8 related legal cases.) since there were less than 10 results for these sentences. The data collection process for the retrieval dataset is visualized in Figure 4.2. There are 257 decision texts in the retrieval dataset. The decision texts related to the same sentence were counted as similar legal cases. Since our corpus does not get updated, we neglected the decision date for cases. If a case named A is determined as the prior case to a case named B, then case B is also a prior case of A according to our ground truth labels. Thus, similar cases are the prior cases of each other. The retrieval dataset can be accessed from Github[2]. The purpose of the prior legal case retrieval task is retrieving cases for a query document where the decision texts of the retrieved cases and the query document are similar. Therefore, the methods utilized in the task must process the whole query document and the documents of the legal database of interest.

[2]https://github.com/koc-lab/yargitay_retrieval_dataset

Table 4.2: Sentences searched on the website of the Court of Cassation to collect legal cases from the civil divisions of the Turkish Court of Cassation.

Original Sentence (Turkish)	**Translation to English**
Dava konusu taşınmaz fiilen bölünerek kullanılmaktadır.	The real estate subject to the case is being used as de facto divided.
Boşanmak isteyen eş diğer eşten daha fazla kusurludur.	The spouse claiming divorce is more at fault than the other spouse.
Mahkemece belirlenen nafaka miktarı yetersizdir.	The amount of alimony determined by the court is insufficient.
Davalının taşınmazı iyiniyetle edindiğinin kabulü gerekir.	It must be accepted that the defendant acquired the real estate in good faith.
Dava konusu işlem muvazaalıdır.	The transaction subject to the case is simulated.
Taşınmaz aile konutu niteliğindedir.	The real estate is a family residence.
Sözleşmede yer alan hüküm haksız şart niteliğindedir.	The provision in the contract is an unfair term.
Dava konusu olayda uyarlamanın koşulları oluşmuştur.	The conditions for the adaptation are met in the case.
Mahkemece hükmedilen manevi tazminat miktarı çok yüksektir.	The amount of immaterial compensation awarded by the court is very high.
Davacının kişilik hakları ihlal edilmiştir.	Personal rights of the plaintiff have been violated.
Bu davranış hakkın kötüye kullanılması niteliğindedir.	This act constitutes an abuse of right.
Tasarrufun saklı pay kurallarını ihlal etmek amacıyla yapıldığı anlaşılmaktadır.	It is understood that the disposition is made to violate the rules on reserved share.
Anonim şirket genel kurul kararı geçersizdir.	The decision of the general stockholders meeting of the joint stock company is invalid.

Table 4.3: Sentences searched on the website of the Court of Cassation to collect legal cases from the criminal divisions of the Turkish Court of Cassation.

Original Sentence (Turkish)	**Translation to English**
Olayda dolandırıcılık suçunun unsurları oluşmuştur.	Elements of fraud exist in the incident.
Sanık suçu haksız tahrik koşulları altında işlemiştir.	The accused committed the crime under the conditions of unjust provocation.
Sanık lehine mevcut takdiri indirim sebepleri uygulanmamıştır.	Existing discretionary reduction grounds in favor of the accused have not been applied.
Sanığın eylemi ifade özgürlüğünün sınırları içinde kalmaktadır.	The act of the accused falls within the limits of freedom of expression.
Maktulün elbisesinde kan izlerine rastlanmıştır.	Traces of blood were found on the victim's clothing.
Sanığın cinayeti tasarlayarak işlediği anlaşılmaktadır.	It is understood that the accused committed a planned murder.
Sanık haberleşmenin gizliliğini ihlal etmiştir.	The accused violated the confidentiality of communications.
Sanığın eylemi meşru müdafaa niteliğindedir.	The act of the accused constitutes self-defense.
Sahte belgenin aldatma yeteneği bulunmaktadır.	Forged document has the ability to deceive.
Sanık suçu olası kastla işlemiştir.	The accused committed the crime with probable intent
Sanığın cezai ehliyeti bulunmamaktadır.	The accused has no criminal capacity.
Sanığın mağdura gönderdiği mesajlar cinsel taciz niteliğindedir.	Messages sent by the accused to the victim constitute sexual harassment.
Sanığın eylemi nitelikli hırsızlık olarak değerlendirilmelidir.	The act of the accused should be considered as qualified theft.

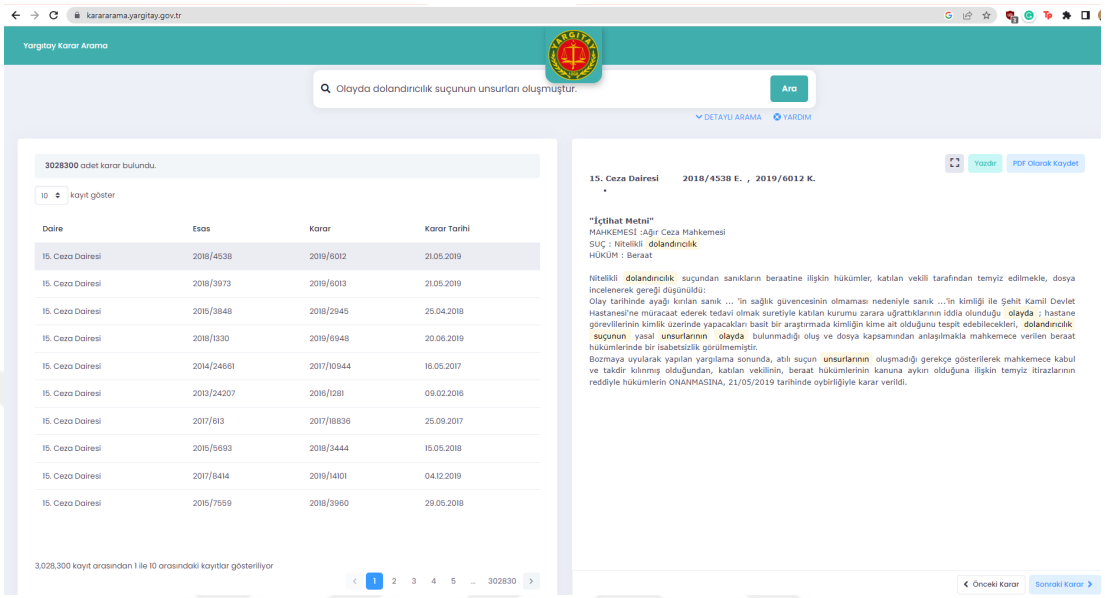


Figure 4.1: A screenshot showing how legal cases related to the sentence "Olayda dolandırıcılık suçunun unsurları oluşmuştur." given in Table 4.3 are found on the official website of the Court of Cassation.

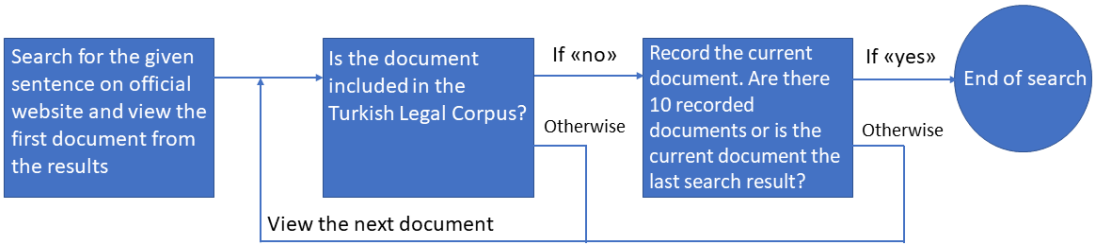


Figure 4.2: Visualization of the data collection process for the retrieval dataset.

We evaluate the performance of our retrieval methods with the micro-precision@k, micro-recall@k, and micro-F1@k metrics. Micro-precision@k is defined for an information retrieval (IR) algorithm as the number of correctly retrieved documents divided by the number of total candidate documents. We assume that the retrieval algorithm finds k candidate documents when we evaluate the micro-precision@k score. The formula for the micro-precision@k metric is provided below for an IR algorithm that assigns the numbers 1, 2, ..., N to the candidate documents such that more similar documents are assigned to smaller

numbers:

$$\text{Micro-precision@k (\%)} = 100 \times \frac{\sum_{i=1}^{N}\sum_{j=1}^{k} a_{ij}}{N \times k},$$
$$a_{ij} = \begin{cases} 1, & \text{if the document j matches with the query document i,} \\ 0, & \text{otherwise,} \end{cases} \tag{4.1}$$

where N is the total number of documents. Micro-precision@k gives the same weight to all samples. Micro-recall@k is the number of correctly retrieved documents divided by the number of total related documents. Micro-recall@k is evaluated as follows:

$$\text{Micro-recall@k (\%)} = 100 \times \frac{\sum_{i=1}^{N}\sum_{j=1}^{k} a_{ij}}{\sum_{i=1}^{N} r_i}. \tag{4.2}$$

where r_i is the number of related documents for the query document i and a_{ij} is defined in Eq. 4.1. Then, micro-F1 is found as follows:

$$\text{Micro-F1@k (\%)} = \frac{2 \times \text{Micro-precision@k} \times \text{Micro-recall@k}}{\text{Micro-precision@k} + \text{Micro-recall@k}}. \tag{4.3}$$

Chapter 5

Prior Legal Case Retrieval with Recurrent Neural Network Autoencoders

In this chapter, we describe our retrieval methods involving RNNs. We have three methods explained in the sections below. We utilize the Turkish word embeddings of [69] in all methods described in this chapter to represent the words of the decision texts. [69] utilized the word2vec CBOW algorithm [63] to train these embeddings. These embeddings were trained on Turkish Wikipedia articles. Thus, they are referred to as Wikipedia embeddings in this thesis. All retrieval methods involving RNNs were trained on the decision texts of the Turkish Legal Corpus that belong to the Turkish Court of Cassation. 80%, 10%, and 10% of the decision texts were used as the training, validation, and test sets for each autoencoder model, respectively.

5.1 Vanilla RNN Autoencoders

In our first method, a representation vector was created for each legal case. Then, the similarity between these vectors was checked to find out if there was a relation between cases. We decided to use a self-supervised algorithm to create the vectors because the retrieval dataset was not large enough to use for training a fully supervised deep learning model. Therefore, the retrieval dataset was used only for testing our models, and autoencoder models were used to produce the vectors. A simplified architecture of our autoencoder models is presented in Figure 5.1. The autoencoder model takes the tokens of the legal case decision text as the input. Then, it encodes the word vectors of the tokens using an encoder RNN model. The output of the encoder, i.e., the hidden state of the encoder, is the desired text vector. Then, this vector is processed by the decoder RNN model to reconstruct the word vectors of the tokens. L2 loss was the preferred metric to compare the original word embeddings with the reconstructed word embeddings. Then, the gradients of this loss were computed to train the autoencoder model. GRU and LSTM were used asl the encoder and decoder models. After the training of the autoencoder models, the cases in the retrieval dataset were fed into them to produce their document vectors. Finally, these vectors were used in retrieval. For each case in the retrieval dataset, other cases were ranked based on the cosine similarity scores between the embeddings of the inspected case and the other cases.

Decision texts of the Court of Cassation are long documents. Long documents can be a problem for RNN variants because the RNNs could forget the start of a long document before they process the end of the document. Therefore, in this method, we divided each decision text into chunks with 100 words. Then, we trained and tested the autoencoder model with these chunks. During retrieval, the average of the encoding vector of each chunk of a document was used as the encoding vector of the document.

We now describe how the vanilla GRU and LSTM models process the Wikipedia word embeddings to generate their hidden states. The input Wikipedia

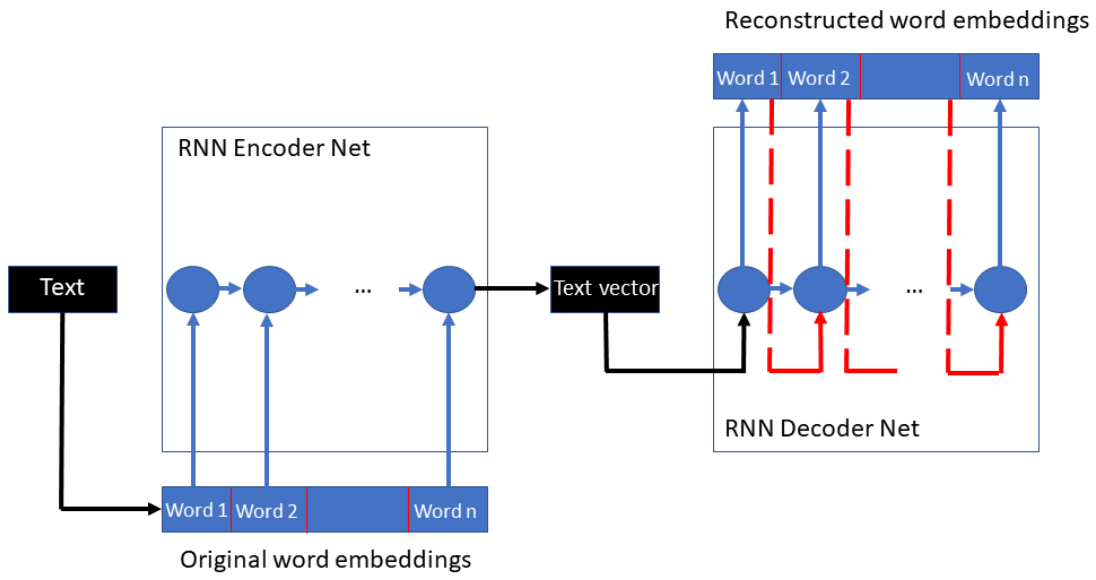


Figure 5.1: Autoencoder architecture.

vector $\boldsymbol{x}_t$ is processed by GRU to generate the hidden state vector $\boldsymbol{h}_t$. The process of mapping the input vector $\boldsymbol{x}_t$ into the hidden state $\boldsymbol{h}_t$ is executed in GRU as follows:

$$\begin{aligned}
\boldsymbol{z}_t &= \sigma(\boldsymbol{W}_z[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_z), \\
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_r), \\
\tilde{\boldsymbol{h}}_t &= \tanh(\boldsymbol{W}_h[\boldsymbol{x}_t; \boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}] + \boldsymbol{b}_h), \\
\boldsymbol{h}_t &= (\boldsymbol{1} - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t,
\end{aligned} \tag{5.1}$$

where the sigmoid activation function, the weight matrix of gate k, and the bias vector of gate k are denoted as $\sigma(.)$, $\boldsymbol{W}_k$, and $\boldsymbol{b}_k$, respectively. Likewise, LSTM processes the input vector $\boldsymbol{x}_t$ to produce the hidden state $\boldsymbol{h}_t$ as follows:

$$\begin{aligned}
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}_f[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_f), \\
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}_i[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_i), \\
\tilde{\boldsymbol{c}}_t &= \tanh(\boldsymbol{W}_c[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_c), \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t, \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}_o[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_o), \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t).
\end{aligned} \tag{5.2}$$

5.2 Combination of the RNN Autoencoders with BM25 Algorithms

In the second method, the decision texts are first ranked with the BM25 algorithms [70]. Then, the first 20 decision texts are re-ranked with RNN autoencoders. This approach takes both exact matches between documents and semantic similarities of the documents into account. Retrieval algorithms rank the documents according to their retrieval status values (RSV). The inverse document frequency (IDF) term of RSV formulas can differ in different implementations of the same BM25 algorithm. We used an open-source implementation[1] of the BM25 algorithms. The following subsections explain the BM25 algorithms used in this work. We utilize the same loss function as the first method, and we divide each decision text into chunks with 100 words again.

5.2.0.1 ATIRE BM25

The ATIRE BM25 algorithm [71] is a variation of Okapi BM25 [70], the first BM25 algorithm. It was used in the ATIRE search engine. A document's RSV is calculated using our implementation of the ATIRE BM25 algorithm as follows:

$$RSV_d = \sum_{t \in q} (log(\frac{N - df_t + 0.5}{df_t + 0.5}) \times \frac{(k_1 + 1) tf_{td}}{k_1((1 - b) + b(\frac{L_d}{L_{avg}})) + tf_{td}}), \tag{5.3}$$

where the count of documents is N, the count of documents containing the term t is df_t, the frequency of the term t in document d is tf_{td}, the length of document d is L_d, the average length of the documents is L_{avg}, $k_1 = 0.9$, and $b = 0.4$ [71].

[1] https://github.com/dorianbrown/rank_bm25

5.2.0.2 BM25L

The BM25L algorithm [72] is another variation of the Okapi BM25 algorithm. In [72], it is mentioned that Okapi BM25 is unsuitable for long documents, and BM25L was created by changing Okapi BM25 for use in long documents. Since the documents in our retrieval dataset were long, we experimented with BM25L. RSV of a document is found by our implementation of the BM25L algorithm as follows:

$$RSV_d = \sum_{t \in q} (log(\frac{N - df_t + 0.5}{df_t + 0.5}) \times (k_1 + 1) \times \frac{c(q,d) + \delta}{k_1 + c(q,d) + \delta}), \tag{5.4}$$

$$c(q,d) = \frac{df_t}{1 - b + b \times \frac{L_d}{L_{avg}}}, \tag{5.5}$$

where the count of documents is N, the count of documents containing the term t is df_t, the frequency of the term t in document d is tf_{td}, the length of document d is L_d, the average length of the documents is L_{avg}, $k_1 = 1.5$, $b = 0.75$, and $\delta = 0.5$ [72].

5.3 Legal RNN Autoencoders

In our third method, we propose a legal word embedding model named LawTurk2Vec to represent the legal words better. We trained the LawTurk2Vec embeddings with the word2vec CBOW algorithm. LawTurk2Vec was trained on the Turkish Legal Corpus, which was introduced in Chapter 4.

Inspired from [73], we propose two legal knowledge incorporation methods, Legal Gate and Legal Cell, to create Legal RNN autoencoders as a prior legal case retrieval method for the Turkish Court of Cassation. Legal Gate and Legal Cell are described in Sections 5.3.1 and 5.3.2 below. Our third method is the same as our first method with an exception: we replace vanilla RNN encoders of

the RNN autoencoders with Legal RNNs to incorporate the LawTurk2Vec word embeddings. On the other hand, we utilize the same loss function as the first method, and we divide each decision text into chunks with 100 words as well.

5.3.1 Legal Gate

This method adds a legal output gate to the standard RNN cell. This additional gate determines the amount of legal embedding knowledge that is injected into the hidden state of the cell. Also, the legal vector (lawTurk2Vec vector) of the input word is merged with the original hidden state, and the resulting vector replaces the hidden state in the equations that find the outputs of the gates. We propose the Legal RNN+Gate models based on the Legal Gate method. The standard GRU gating equations given in Eq. 5.1 were modified to create the Legal GRU+Gate model, which is a Legal RNN+Gate model variant. The gating equations of the Legal GRU+Gate model are given as follows:

$$
\begin{aligned}
\boldsymbol{z}_t &= \sigma(\boldsymbol{W}_z[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \underline{\boldsymbol{\pi}_t}] + \boldsymbol{b}_z), \\
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \underline{\boldsymbol{\pi}_t}] + \boldsymbol{b}_r), \\
\underline{\boldsymbol{o}_t^L} &= \underline{\sigma(\boldsymbol{W}_r[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \boldsymbol{\pi}_t] + \boldsymbol{b}_o)}, \\
\tilde{\boldsymbol{h}}_t &= \tanh(\boldsymbol{W}_h[\boldsymbol{x}_t; \boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}] + \boldsymbol{b}_h), \\
\boldsymbol{h}_t &= (\boldsymbol{1} - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t + \underline{\boldsymbol{o}_t^L \odot \tanh(\boldsymbol{\pi}_t)},
\end{aligned}
\tag{5.6}
$$

where the legal vector for the input word is denoted as $\boldsymbol{\pi}_t$ and modified parts of the equations are underlined. Following the same procedure used for Legal GRU+Gate, the standard LSTM gating equations given in Eq. 5.2 were modified as follows to create the Legal LSTM+Gate model, which is another Legal

RNN+Gate variant:

$$
\begin{aligned}
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}_f[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \underline{\boldsymbol{\pi}_t}] + \boldsymbol{b}_f), \\
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}_i[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \underline{\boldsymbol{\pi}_t}] + \boldsymbol{b}_i), \\
\tilde{\boldsymbol{c}}_t &= \tanh(\boldsymbol{W}_c[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_c), \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t, \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}_o[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \underline{\boldsymbol{\pi}_t}] + \boldsymbol{b}_o), \\
\underline{\boldsymbol{o}_t^L} &= \underline{\sigma(\boldsymbol{W}_o[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \boldsymbol{\pi}_t] + \boldsymbol{b}_o)}, \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t) + \underline{\boldsymbol{o}_t^L \odot \tanh(\boldsymbol{W}_L[\boldsymbol{\pi}_t])},
\end{aligned} \tag{5.7}
$$

where the modified parts of the equations are underlined.

5.3.2 Legal Cell

This method processes legal vectors with a standard RNN model. Then, another RNN model processes daily language vectors. However, this model takes the outputs of the standard RNN model as inputs to its gates. Based on the Legal Cell method, we propose the Legal RNN+Cell models. According to the method, the gating equations for Legal GRU+Cell, which is a Legal RNN+Cell model, become:

$$
\begin{aligned}
\boldsymbol{h}_t^L &= GRU(\boldsymbol{\pi}_t) \\
\boldsymbol{z}_t &= \sigma(\boldsymbol{W}_z[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \boldsymbol{h}_t^L] + \boldsymbol{b}_z) \\
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r[\boldsymbol{x}_r; \boldsymbol{h}_{t-1}; \boldsymbol{h}_t^L] + \boldsymbol{b}_r) \\
\tilde{\boldsymbol{h}}_t &= \tanh(\boldsymbol{W}_r[\boldsymbol{x}_t; \boldsymbol{r}_t \odot (\boldsymbol{h}_{t-1} + \boldsymbol{h}_t^L)] + \boldsymbol{b}_h) \\
\boldsymbol{h}_t &= (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \tilde{\boldsymbol{h}}_t,
\end{aligned} \tag{5.8}
$$

where $\boldsymbol{h}_t^L$ is the hidden state of Legal GRU+Cell. Likewise, the parts of the input vector, $\boldsymbol{x}_t$, and $\boldsymbol{\pi}_t$, are processed by the Legal LSTM+Cell model, which is another Legal RNN+Cell model as follows:

$$
\begin{aligned}
\boldsymbol{c}_t^L, \boldsymbol{h}_t^L &= LSTM(\boldsymbol{\pi}_t) \\
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}_f[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}] + \boldsymbol{b}_f) \\
\boldsymbol{f}_t^L &= \sigma(\boldsymbol{W}_f^L[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}^L] + \boldsymbol{b}_f^L) \\
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}_i[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \boldsymbol{h}_t^L] + \boldsymbol{b}_i) \\
\tilde{\boldsymbol{c}}_t &= \tanh(\boldsymbol{W}_c[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \boldsymbol{h}_t^L] + \boldsymbol{b}_c) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{f}_t^L \odot \boldsymbol{c}_t^L + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}_o[\boldsymbol{x}_t; \boldsymbol{h}_{t-1}; \boldsymbol{h}_t^L] + \boldsymbol{b}_o) \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t),
\end{aligned} \tag{5.9}
$$

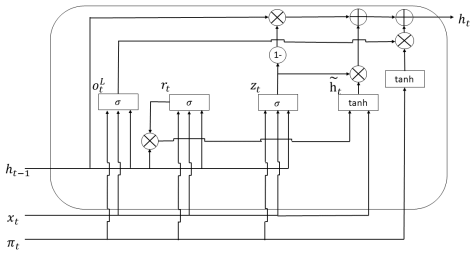
where $\boldsymbol{c}_t^L$, and $\boldsymbol{h}_t^L$ are the cell state, and hidden state of the Legal LSTM+Cell, respectively.

5.4 Performance of the RNN Autoencoders

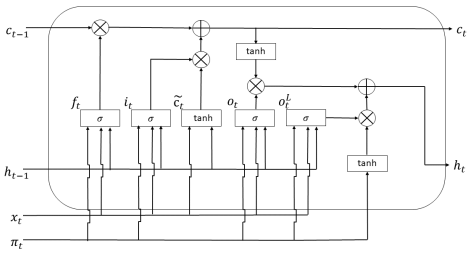
In Table 5.1, the results of the experiments with the RNN autoencoders for the Court of Cassation are shown. All models were tested on the retrieval dataset introduced in Chapter 4. We first implemented our first two retrieval methods involving RNN autoencoders. After observing that combining RNN autoencoders with ATIRE BM25 provides better results than the other implemented models, we decided to experiment with the combination of Legal RNN autoencoders with ATIRE BM25 while testing the trained Legal RNN autoencoders as well. Upon observing Table 5.1, the best-performing method for the prior case retrieval task is the proposed LSTM & ATIRE BM25 model. Autoencoders with Legal RNNs do not use the same RNN model for their encoder and decoder parts. That is, they use Legal RNNs as encoders and vanilla RNNs as decoders. The difference between the architectures of the encoder and decoder models might be the reason why autoencoders with Legal RNN autoencoders have a poorer performance than vanilla RNN autoencoders on the prior case retrieval task.

Table 5.1: Performance of RNN autoencoders on the retrieval dataset. Each row represents an autoencoder model containing the written RNN model as its encoder model. Scores are percentages. The best scores are emboldened.

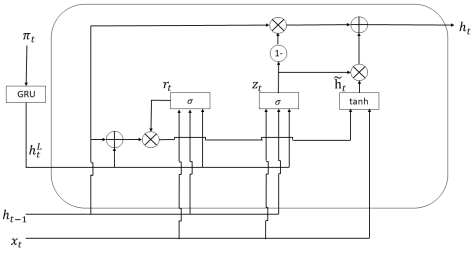
Method	**Micro-F1@5**	**Micro-Precision@5**	**Micro-Recall@5**
GRU	16.96	18.05	15.99
LSTM	14.26	14.86	13.70
GRU & ATIRE BM25	38.69	39.30	38.09
LSTM & ATIRE BM25	**39.98**	**41.09**	38.94
GRU & BM25L	37.94	38.13	37.75
LSTM & BM25L	38.70	38.37	**39.04**
Legal GRU+Gate	11.84	12.51	11.24
Legal LSTM+Gate	5.32	5.68	5.00
Legal GRU+Cell	20.05	21.63	18.68
Legal LSTM+Cell	11.80	12.14	11.47
Legal GRU+Gate & ATIRE BM25	20.31	20.65	19.97
Legal LSTM+Gate & ATIRE BM25	19.60	19.86	19.35
Legal GRU+Cell & ATIRE BM25	39.24	39.46	39.03
Legal LSTM+Cell & ATIRE BM25	37.51	37.67	37.35



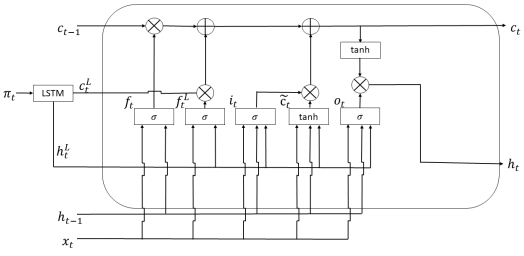
(a) Legal GRU+Gate



(b) Legal LSTM+Gate



(c) Legal GRU+Cell



(d) Legal LSTM+Cell

Figure 5.2: Legal RNN Architectures.

Chapter 6

Prior Legal Case Retrieval with Transformers

6.1 BERTurk-Legal

In this chapter, we introduce our BERTurk-Legal model. We create BERTurk-Legal by further pre-training the BERTurk model [74] on the Turkish Legal Corpus. Therefore, BERTurk-Legal and BERTurk have the same model architecture.

BERTurk was trained on several large Turkish corpora [74] and has the same architecture as the BERT-BASE [47] model, which is one of the most influential transformer models. Thus, BERT-BASE and BERTurk-Legal have the same architecture. BERTurk-Legal has 12 layers, a maximum number of tokens for each input of 512, and an embedding size of 768. It truncates long texts with more than 512 tokens. Each layer of BERTurk-Legal consists of two sublayers in serial connection. The output of the first sublayer is found as follows:

$$\boldsymbol{O_1} = \text{Layer Norm}(\boldsymbol{X} + \text{Self Attention}(\boldsymbol{X})). \tag{6.1}$$

In the equation above, $\boldsymbol{O_1}$ is the output, and $\boldsymbol{X}$ is the input embeddings [46]. The Self Attention layer has h heads and the output of the Self Attention layer

is found as follows:

$$\begin{aligned} \text{Self Attention}(\boldsymbol{X}) &= \text{Concatenate}(\text{Head}_{\boldsymbol{1}}, ..., \text{Head}_{\boldsymbol{h}})\boldsymbol{W^O}, \\ \text{Head}_{\boldsymbol{i}} &= \text{Attention}(\boldsymbol{XW_i^Q}, \boldsymbol{XW_i^K}, \boldsymbol{XW_i^V}), \\ \text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) &= \text{Softmax}(\frac{\boldsymbol{QK}^T}{\sqrt{d_k}})\boldsymbol{V}. \end{aligned} \tag{6.2}$$

In the equation above, $h = 12$, $\boldsymbol{W_i^O}$, $\boldsymbol{W_i^Q}$, $\boldsymbol{W_i^K}$, and $\boldsymbol{W_i^V}$ are weight matrices, d_k is a trainable parameter, and $\boldsymbol{K}^T$ is the transpose of $\boldsymbol{K}$ [46, 47]. The output of the second sublayer is found as follows:

$$\boldsymbol{O_2} = \text{Layer Norm}(\boldsymbol{O_1} + \text{FFNN}(\boldsymbol{O_1})). \tag{6.3}$$

In the equation above, $\boldsymbol{O_2}$ is the output of a layer of BERTurk-Legal. *FFNN* is an FFNN with two linear layers and ReLU activation between these layers [46].

The initial values of the parameters of BERTurk-Legal are the same as the trained values of the parameters of BERTurk. Thus, the knowledge from the daily Turkish language has been transferred to the BERTurk-Legal model. Turkish corpora used to train BERTurk consist of texts written in daily language, and the total size of the corpora is 35 GB [74]. We further pre-trained BERTurk-Legal on the decision texts of the Turkish Legal Corpus that belong to the Turkish Court of Cassation. 80%, 10%, and 10% of the decision texts were used as the training, validation, and test sets for the pre-training process. Since BERTurk was trained on texts with only lowercase letters, we converted our decision texts to lowercase before utilizing them for training. We used masked language modeling (MLM) as our pre-training method. We masked 20% of the words in the training dataset, and the BERTurk-Legal model was trained by predicting the masked words.

We utilize BERTurk-Legal to create document vectors of the decision texts of the retrieval dataset introduced in Chapter 4. The output embedding vector for the first token of the input text is used as the document vector of the input text. The first token is a special token named CLS, which is added to the start of every input before a BERT-based model processes the input. The CLS embedding contains information related to the whole input text [47]. Then, BERTurk-Legal retrieves prior legal cases by ranking the candidate documents according to the

cosine similarities between the document vectors of the candidate documents and the query document. The retrieval process is visualized in Figure 6.1.

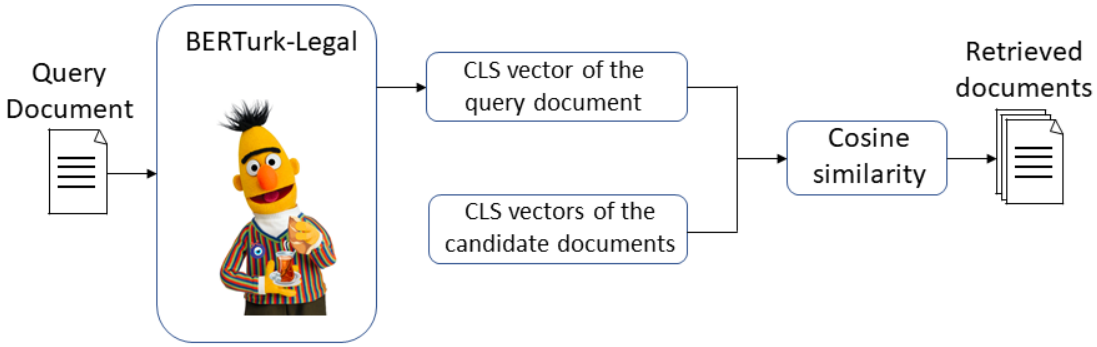


Figure 6.1: Overview of how BERTurk-Legal is utilized to retrieve documents.

6.2 Performance of BERTurk-Legal

In this section, we compare the performance of BERTurk-Legal with BERTurk and our best-performing RNN autoencoder method, LSTM & ATIRE BM25. We trained these models on the same training dataset as before with three seed values to generate three instances of these models so that the comparison is more reliable. We present the result in Table 6.1. The table shows the average scores of the instances of the models. We compare BERTurk with BERTurk-Legal to show that our further pre-training approach benefits the prior legal case retrieval task. Our results show that BERTurk-Legal is the best-performing model in all evaluation metrics. BERTurk-Legal outperforms BERTurk by a large margin in all metrics, as expected. This difference shows that legal NLP tasks are hard for the language models trained on daily language.

BERTurk-Legal performs better than the LSTM & ATIRE BM25 model according to all evaluation metrics. An important reason for the success of BERTurk-Legal is transfer learning. The LSTM & ATIRE BM25 model did not utilize any machine learning model to initialize its parameters. However, BERTurk-Legal transferred Turkish knowledge from BERTurk. Therefore, BERTurk-Legal has both general and domain-specific expertise on Turkish

Table 6.1: Performance of the language models on the retrieval dataset. Scores are percentages. The best scores are emboldened.

Method	**Micro-F1@5**	**Micro-Precision@5**	**Micro-Recall@5**
LSTM & ATIRE BM25	33.96	47.21	26.52
BERTurk	18.87	26.17	14.76
BERTurk-Legal	**38.14**	**52.89**	**29.82**

texts. Another important reason for the success of BERTurk-Legal is that it is a language model with much more parameters than the LSTM & ATIRE BM25 model.

Chapter 7

Discussion and Conclusions

In this thesis, we introduce the prior legal case retrieval task for the Turkish Court of Cassation. Transformer-based language models are widely used in NLP works in literature. Thus, we introduce the BERTurk-Legal model as a method to solve the prior legal case retrieval task for the Turkish Court of Cassation. We also compare BERTurk-Legal with other methods we experimented on throughout the Master of Science studies and show that it provides state-of-the-art results. BERTurk-Legal was trained on Turkish Legal Corpus. We utilized masked language modeling for training BERTurk-Legal in a self-supervised manner. We expect future works on the ML tasks of the Turkish legal domain to utilize transformer-based models frequently. As computer computing capabilities and memories grow, transformer-based models that allow long input texts with more than 512 tokens will become more popular in literature. On the other hand, hierarchical transformers, transformer-based models that create chunks of text with 512 tokens, process these chunks, and aggregate the results for each chunk, can also be utilized in legal NLP.

Sentence transformer [75] was introduced as an architecture to create document vectors. Sentence transformers use contrastive learning to make their output embeddings more suitable than vanilla BERT models for measuring the similarity between them. Thus, they can be useful in prior legal case retrieval tasks. We

plan to use sentence transformer-based models and hierarchical transformers for legal NLP in our future works.

Bibliography

[1] Y. Ma, Y. Shao, B. Liu, Y. Liu, M. Zhang, and S. Ma, "Retrieving Legal Cases from a Large-scale Candidate Corpus," in *Proceedings of the Eigth International Competition on Legal Information Extraction/Entailment (COLIEE 2021)*, COLIEE '21, pp. 38–42, 2021.

[2] C. E. Öztürk, Ö. Köksal, Ş. B. Özçelik, and A. Koç, "Prior Case Retrieval for the Court of Cassation of Turkey," in *2022 16th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*, pp. 539–544, 2022.

[3] C. E. Öztürk, Ş. B. Özçelik, and A. Koç, "Predicting Outcomes of the Court of Cassation of Turkey with Recurrent Neural Networks," in *2022 30th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2022.

[4] A. C. Aras, C. E. Öztürk, and A. Koç, "Feedforward Neural Network Based Case Prediction in Turkish Higher Courts," in *2022 30th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2022.

[5] E. Mumcuoğlu, C. E. Öztürk, H. M. Ozaktas, and A. Koç, "Natural language processing in law: Prediction of outcomes in the higher courts of Turkey," *Information Processing & Management*, vol. 58, no. 5, 2021.

[6] H. Ye, X. Jiang, Z. Luo, and W. Chao, "Interpretable Charge Predictions for Criminal Cases: Learning to Generate Court Views from Fact Descriptions," 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), pp. 1854–1864, Association for Computational Linguistics, June 2018.

[7] I. Chalkidis, I. Androutsopoulos, and A. Michos, "Obligation and prohibition extraction using hierarchical RNNs," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, (Melbourne, Australia), pp. 254–259, Association for Computational Linguistics, July 2018.

[8] C. Çetindağ, B. Yazıcıoğlu, and A. Koç, "Named-entity recognition in Turkish legal texts," *Natural Language Engineering*, vol. 29, no. 3, p. 615642, 2023.

[9] A. Kanapala, S. Pal, and R. Pamula, "Text summarization from legal documents: A survey," *Artificial Intelligence Review*, vol. 51, pp. 371–402, 2019.

[10] G. Salton and M. J. McGill, *Introduction to modern information retrieval*. McGraw-Hill, Inc., 1986.

[11] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *NeurIPS*, 2013.

[12] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1532–1543, Association for Computational Linguistics, Oct. 2014.

[13] I. Iacobacci, M. T. Pilehvar, and R. Navigli, "Embeddings for Word Sense Disambiguation: An Evaluation Study," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 897–907, Aug. 2016.

[14] L.-C. Yu, J. Wang, K. R. Lai, and X. Zhang, "Refining Word Embeddings for Sentiment Analysis," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Copenhagen, Denmark), pp. 534–539, Sep. 2017.

[15] K. Shuang, R. Li, M. Gu, J. Loo, and S. Su, "Major-Minor Long Short-Term Memory for Word-Level Language Model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 10, pp. 3932–3946, 2020.

[16] H. Ennajari, N. Bouguila, and J. Bentahar, "Combining Knowledge Graph and Word Embeddings for Spherical Topic Modeling," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2021.

[17] N. Gong, N. Yao, and S. Guo, "Seeds: Sampling-Enhanced Embeddings," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 2, pp. 577–586, 2022.

[18] I. Chalkidis and D. Kampas, "Deep learning in law: early adaptation and legal word embeddings trained on large corpora," *Artificial Intelligence and Law*, vol. 27, no. 2, pp. 171–198, 2019.

[19] A. Vaswani *et al.*, "Attention is all you need," in *NeurIPS*, vol. 30, 2017.

[20] M. Peters *et al.*, "Deep contextualized word representations," in *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, (New Orleans, Louisiana), pp. 2227–2237, Jun. 2018.

[21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, MN, USA), pp. 4171–4186, Jun. 2019.

[22] T. Brown *et al.*, "Language models are few-shot learners," in *NeurIPS*, vol. 33, pp. 1877–1901, 2020.

[23] K. D. Ashley, *Modelling legal argument: Reasoning with cases and hypotheticals*. PhD thesis, University of Massachusetts, USA, 1988. Order No: GAX88-13198.

[24] K. D. Ashley, "Reasoning with cases and hypotheticals in HYPO," *International Journal of Man-Machine Studies*, vol. 34, no. 6, pp. 753 – 796, 1991.

[25] C. D. Hafner and D. H. Berman, "The role of context in case-based legal reasoning: Teleological, temporal, and procedural," *Artificial Intelligence and Law*, vol. 10, no. 13, p. 1964, 2002.

[26] A. Wyner, "An ontology in OWL for legal case-based reasoning," *Artificial Intelligence and Law*, vol. 16, no. 361, 2008.

[27] V. Aleven, "Using background knowledge in case-based legal reasoning: A computational model and an intelligent learning environment," *Artificial Intelligence*, vol. 150, pp. 183–237, 11 2003.

[28] K. D. Ashley and S. Brüninghaus, "Automatically classifying case texts and predicting outcomes," *Artificial Intelligence and Law*, vol. 17, no. 2, pp. 125–165, 2009.

[29] T. Bench-Capon, A. M. Araszkiewicz, A. K. Ashley, K. Atkinson, F. Bex, F. Borges, D. Bourcier, P. Bourgine, J. G. Conrad, E. Francesconi, T. F. Gordon, G. Governatori, J. L. Leidner, D. D. Lewis, R. P. Loui, L. T. McCarty, H. Prakken, F. Schilder, E. Schweighofer, P. Thompson, A. Tyrrell, B. Verheij, D. N. Walton, and A. Z. Wyner, "A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law," *Artificial Intelligence and Law*, vol. 20, pp. 215–319, 2012.

[30] A. D. Martin, K. M. Quinn, T. W. Ruger, and P. T. Kim, "Competing approaches to predicting supreme court decision making," *Perspectives on Politics*, vol. 2, no. 4, pp. 761–767, 2004.

[31] T. W. Ruger, P. T. Kim, A. D. Martin, and K. M. Quinn, "The supreme court forecasting project: Legal and political science approaches to predicting supreme court decisionmaking," *Columbia Law Review*, pp. 1150–1210, 2004.

[32] N. Aletras, D. Tsarapatsanis, D. Preoiuc-Pietro, and V. Lampos, "Predicting judicial decisions of the European Court of Human Rights: a Natural Language Processing perspective," *PeerJ Computer Science*, vol. 2, 2016.

[33] D. M. Katz, M. J. Bommarito, and J. Blackman, "A general approach for predicting the behavior of the supreme court of the United States," *PloS one*, vol. 12, no. 4, p. e0174698, 2017.

[34] H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, and M. Sun, "Legal Judgment Prediction via Topological Learning," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, (Brussels, Belgium), pp. 3540–3549, Association for Computational Linguistics, Oct.-Nov. 2018.

[35] W. Yang, W. Jia, X. Zhou, and Y. Luo, "Legal Judgment Prediction via Multi-Perspective Bi-Feedback Network," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 4085–4091, International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[36] M. B. Virtucio, J. Aborot, J. K. Abonita, R. Aviñante, R. J. Copino, M. Neverida, V. Osiana, E. Peramo, J. Syjuco, and G. B. Tan, "Predicting decisions of the Philippine supreme court using natural language processing and machine learning," in *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 2, pp. 130–135, 2018.

[37] B. Strickson and B. De La Iglesia, "Legal Judgement Prediction for UK Courts," in *Proceedings of the 2020 The 3rd International Conference on Information Science and System*, ICISS 2020, (New York, NY, USA), p. 204209, Association for Computing Machinery, 2020.

[38] J. Niklaus, I. Chalkidis, and M. Stürmer, "Swiss-Judgment-Prediction: A Multilingual Legal Judgment Prediction Benchmark," in *Proceedings of the Natural Legal Language Processing Workshop 2021*, (Punta Cana, Dominican Republic), pp. 19–35, Association for Computational Linguistics, Nov. 2021.

[39] N. Aletras, D. Tsarapatsanis, D. Preoţiuc-Pietro, and V. Lampos, "Predicting judicial decisions of the European Court of Human Rights: A natural language processing perspective," *PeerJ Computer Science*, vol. 2, p. e93, 2016.

[40] C. O'Sullivan and J. Beel, "Predicting the Outcome of Judicial Decisions made by the European Court of Human Rights," in *Proceedings of the 27th AIAI Irish Conference on Artificial Intelligence and Cognitive Science*, (Dublin, Ireland), 2019.

[41] M. Medvedeva, M. Vols, and M. Wieling, "Using machine learning to predict decisions of the European Court of Human Rights," *Artificial Intelligence and Law*, vol. 28, pp. 237–266, June 2020.

[42] I. Chalkidis, I. Androutsopoulos, and N. Aletras, "Neural Legal Judgment Prediction in English," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 4317–4323, Association for Computational Linguistics, July 2019.

[43] I. Chalkidis, E. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "Extreme Multi-Label Legal Text Classification: A Case Study in EU Legislation," in *Proceedings of the Natural Legal Language Processing Workshop 2019*, (Minneapolis, Minnesota), pp. 78–87, Association for Computational Linguistics, June 2019.

[44] J. Ge, Y. Huang, X. Shen, C. Li, and W. Hu, "Learning Fine-Grained Fact-Article Correspondence in Legal Cases," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 29, p. 36943706, nov 2021.

[45] D. W. Otter, J. R. Medina, and J. K. Kalita, "A Survey of the Usages of Deep Learning for Natural Language Processing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 604–624, 2021.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[47] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, (Minneapolis, Minnesota), pp. 4171–4186, Association for Computational Linguistics, June 2019.

[48] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language Models are Few-Shot Learners," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 1877–1901, Curran Associates, Inc., 2020.

[49] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[50] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "LEGAL-BERT: The Muppets straight out of Law School," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, (Online), pp. 2898–2904, Association for Computational Linguistics, Nov. 2020.

[51] J. Valvoda, R. Cotterell, and S. Teufel, "On the Role of Negative Precedent in Legal Outcome Prediction," 2022.

[52] I. Chalkidis, A. Jana, D. Hartung, M. J. Bommarito, I. Androutsopoulos, D. M. Katz, and N. Aletras, "LexGLUE: A Benchmark Dataset for Legal Language Understanding in English," *Available at SSRN 3936759*, 2021.

[53] J. Niklaus, V. Matoshi, P. Rani, A. Galassi, M. Strmer, and I. Chalkidis, "LEXTREME: A Multi-Lingual and Multi-Task Benchmark for the Legal Domain," 2023.

[54] O. Akça, G. Bayrak, A. M. Issifu, and M. C. Ganiz, "Traditional Machine Learning and Deep Learning-based Text Classification for Turkish Law Documents using Transformers and Domain Adaptation," in *2022 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*, pp. 1–6, 2022.

[55] N. Sevim and A. Koç, "Investigation of Gender Bias in Turkish Word Embeddings," in *29th Signal Processing and Communications Applications Conference, SIU 2021, Istanbul, Turkey, June 9-11, 2021*, pp. 1–4, IEEE, 2021.

[56] N. Sevim, F. Şahinuç, and A. Koç, "Gender bias in legal corpora and debiasing it," *Natural Language Engineering*, vol. 29, no. 2, p. 449482, 2023.

[57] J. Rabelo, R. Goebel, Y. Kano, M.-Y. Kim, M. Yoshioka, and K. Satoh, "Summary of the Competition on Legal Information Extraction/Entailment (COLIEE) 2021," in *Proceedings of the Eigth International Competition on Legal Information Extraction/Entailment (COLIEE 2021)*, COLIEE '21, pp. 1–7, 2021.

[58] V. Tran, M. L. Nguyen, and K. Satoh, "Building legal case retrieval systems with lexical matching and summarization using a pre-trained phrase scoring model," in *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Law*, ICAIL '19, (New York, NY, USA), p. 275282, Association for Computing Machinery, 2019.

[59] S. Althammer, A. Askari, S. Verberne, and A. Hanbury, "DoSSIER@COLIEE 2021: Leveraging dense retrieval and summarization-based re-ranking for case law retrieval," in *Proceedings of the Eigth International Competition on Legal Information Extraction/Entailment (COLIEE 2021)*, COLIEE '21, pp. 8–14, 2021.

[60] B. Ali, R. More, S. Pawar, and G. K. Palshikar, "Prior Case Retrieval using Evidence Extraction from Court Judgements," in *18th International Conference on Artificial Intelligence and Law*, 2021.

[61] S. Althammer, S. Hofstätter, M. Sertkan, S. Verberne, and A. Hanbury, "PARM: A Paragraph Aggregation Retrieval Model for Dense Document-to-Document Retrieval," in *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 1014, 2022, Proceedings, Part I*, (Berlin, Heidelberg), p. 1934, Springer-Verlag, 2022.

[62] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A Neural Probabilistic Language Model," *Journal of Machine Learning Research*, vol. 3, p. 11371155, Mar 2003.

[63] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," in *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2013.

[64] I. Chalkidis, E. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "Extreme Multi-Label Legal Text Classification: A Case Study in EU Legislation," in *Proceedings of the Natural Legal Language Processing Workshop 2019*, (Minneapolis, Minnesota), Association for Computational Linguistics, June 2019.

[65] W. Zhao, G. Zhang, G. Yuan, J. Liu, H. Shan, and S. Zhang, "The Study on the Text Classification for Financial News Based on Partial Information," *IEEE Access*, vol. 8, pp. 100426–100437, 2020.

[66] Y. Kim, "Convolutional Neural Networks for Sentence Classification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1746–1751, ACM, 2014.

[67] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*, (San Diego, CA, USA), 2015.

[68] T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," in *Proceedings of the 2015 Conference*

on Empirical Methods in Natural Language Processing (EMNLP), (Lisbon, Portugal), pp. 1412–1421, Association for Computational Linguistics, Sept. 2015.

[69] A. Köksal, "Turkish pre-trained Word2vec model." `https://github.com/akoksal/Turkish-Word2Vec`, 2018.

[70] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford, "Okapi at trec-3," in *TREC*, 1994.

[71] A. Trotman, X. Jia, and M. Crane, "Towards an efficient and effective search engine," in *OSIR@SIGIR*, 2012.

[72] Y. Lv and C. Zhai, "When documents are very long, bm25 fails!," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, (New York, NY, USA), p. 11031104, Association for Computing Machinery, 2011.

[73] Y. Qin, F. Qi, S. Ouyang, Z. Liu, C. Yang, Y. Wang, Q. Liu, and M. Sun, "Improving Sequence Modeling Ability of Recurrent Neural Networks via Sememes," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2364–2373, 2020.

[74] S. Schweter, "BERTurk - BERT models for Turkish." `https://doi.org/10.5281/zenodo.3770924`, 2020. Accessed:2022-06-19.

[75] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 11 2019.