POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Politecnico di Torino

Master's Degree Thesis

Court Judgment Prediction and Explanation based on Transformers

Supervisors

Prof. Luca CAGLIERO

Dr. Irene BENEDETTO

Candidate

Salvatore Junior CURELLO

December 2023

Summary

Over the last few years, Artificial Intelligence (AI) has attracted much attention as it is becoming increasingly common in our lives. AI involves the development of automated systems capable of performing tasks that typically require humans. It can be applied in a wide range of applications such as healthcare, finance, education, social media, agriculture, etc.

Machine Learning (ML) plays a pivotal role in this context. ML algorithms have the main advantage that they can adapt and evolve based on the information that they analyze.

This thesis is dedicated to exploring the legal domain, focusing on using Machine Learning techniques, particularly those based on Natural Language Processing (NLP) methods. NLP is a sub-field of AI that tries to make an interaction between computers and human language in order to enable machines to understand, interpret, and generate human language.

However, it is important to consider some intrinsic characteristics of this field. The legal domain is complex, and rich in peculiar terms that are very technical and unique to this area. Then, a certain knowledge is required to understand them properly. Unfortunately, common Language Models (LM) face limitations in this context, since they are trained on general language patterns and do not have specific knowledge which is essential to understand and interpret legal concepts. Furthermore, legal documents are typically quite long, verbose, and noisy. The limitations of LM in terms of the amount of data they can process, is another significant challenge in this field.

Thesis Objectives

This thesis focuses on two different but closely related tasks. The first one is the *Court Judgment Prediction* task. It consists of predicting the outcomes of legal cases by analyzing patterns in past cases and identifying relevant factors that may impact the outcome of a court decision. This can be beneficial in many highly populated countries characterized by a vast number of pending legal cases that impede the judicial process due to multiple factors, including the unavailability of

competent judges.

The second part of this thesis, instead, focuses on the *Court Judgment Prediction and Explanation* task. In the legal domain, the mere prediction of a legal case is not sufficient if not accompanied by the corresponding explanation. Then, the goal is to provide a model that not only gives as outcome the final decision of a legal case but also indicates the reasoning behind the predicted decision.

Contributions

The main contribution of this work is the implementation of an automated system that performs:

- A Court Judgment Prediction task in which the primary contribution is an extensive experimentation with various Transformers Models, making a comparison between Generic models and Domain-specific Transformers.

- A Court Judgment Prediction and Explanation task in which the application of the occlusions method and attention mechanisms can help to mitigate the disparity between the explanations generated by the machine and those manually annotated by legal experts.

Methods

All the experiments have been conducted using the ILDC (Indian Legal Documents Corpus) which is composed of a large set of Indian Supreme Court cases, expressed in the English language and annotated with original court decisions ("accepted" v/s "rejected").

Concerning the Court Judgment Prediction task, all the Transformers are trained on the last N tokens of the documents where N is equal to the maximum amount of tokens supported by each model.

In the Court Judgment Prediction and Explanation task, instead, the $ILDC_{expert}$ corpus is used. It consists of 56 documents that are extracted from the Test Set and given to five legal experts who were requested to predict the judgment and mark the sentences that they considered explanatory for their decisions. For the explanation generation, the occlusion method and attention mechanism have been exploited to extract the most relevant sentences from the case description that best justify the final decision. Performance evaluation used a battery of metrics that measure the overlap between the expert annotators' gold explanations and those generated by the machine. These metrics essentially quantify how well a system generates text compared to human standards.

Results

Regarding the Court Judgment Prediction task, domain-specific Transformer models such as LegalLSGBERT, CaseLawBERT, and LegalBERT achieved the best performance in terms of F1-score, outperforming generic models. This demonstrated their effectiveness, even though they were pre-trained on US/EU legal documents with legal systems differing significantly from the Indian context. Furthermore, the application of Hierarchical Transformer models led to improved performances. In particular, the results were slightly higher when using the [CLS] token instead of Mean and Max pooling, proving to be the most effective strategy in almost all models.

In the Court Judgment Prediction and Explanation task, the results obtained using the occlusions method show that, by considering the top 40% of sentences, the baseline is outperformed. Using the attention mechanism, instead, achieved similar performances to the baseline, but the explanations were shorter than the golden annotations of the legal experts. In some cases, this brevity can be considered an advantage, facilitating quick insights to expedite the decision-making process.

Conclusion and future works

Domain-specific Transformer models in Court Judgment Prediction task show effectiveness in different legal systems. Occlusions method and attention mechanisms are crucial for extracting relevant sentences and generate the explanations. Improvements could involve applying Transformer models pretrained on Indian legal documents to capture cultural nuances. Extending hierarchical attention models for datasets with multi-modal information is a potential enhancement for the explanation generation.

Acknowledgements

I express my gratitude to Prof. Luca Cagliero and Dr. Irene Benedetto for allowing me to explore such an interesting topic. Their invaluable guidance and support were crucial in successfully completing this work.

Thank to my family for their unwavering support and efforts that have made this achievement possible. Their encouragement, sacrifices, and love have been instrumental in helping me reach this point.

Special thanks also go to my friends, their uplifting support has not only added an extra layer of joy but has also been a guiding light during both the highs and lows.

Table of Contents

List of Tables

List of Figures

Acronyms

AI
artificial intelligence

NLP
Natural Language Processing

LM
Language Model

ILDC
Indian Legal Documents Corpus

CJPE
Court Judgment Prediction and Explanation

BOW
Bag-of-words

PV-DM
Distributed Memory Model of Paragraph Vectors

PV-DBOW
Distributed Bag of Words of Paragraph Vector

RNN
Recurrent Neural Network

LSTM
Long Short-Term Memory

GRUs

Gated Recurrent Units

BERT

Bidirectional Encoder Representations from Transformers

MLM

Masked Language Model

NSP

Next Sentence Prediction

CBOW

Continuous Bag-of-Words

ML

Machine Learning

TF-IDF

Term Frequency-Inverse Document Frequency

LJP

Legal Judgment Prediction

SVM

Support Vector Machine

LADAM

Law Article Distillation based Attention Network

CJP

Court Judgement Prediction

LED

Longformer-Encoder-Decoder

LSG

Local Sparse Global

Chapter 1

Introduction

In recent years, AI (Artificial Intelligence) has suddenly reshaped the way we live and interact with the world around us. AI involves the development of automated systems capable of performing tasks that typically require humans. These tasks include a wide range of activities such as recognizing patterns, decision-making, and visual perception. The main advantage is that these new methods can be applied to various domains like healthcare, finance, education, social media, agriculture, and beyond.

This thesis specifically directs its focus on the legal domain, in which the vast majority of data in this field are represented in text forms, such as judgment documents, contracts, and legal opinions. The ability of AI to analyze vast amounts of textual information can be a big advantage for legal professionals who invest substantial time in tasks such as legal research, contract analysis, reviewing legal documents, etc.

Machine Learning (ML) plays a pivotal role in many AI applications. ML algorithms have the main advantage that they can adapt and evolve based on the information that they analyze. Most of the ML techniques applied in the legal context are based on Natural Language Processing (NLP) methodologies. NLP is a sub-field of ML that tries to make an interaction between computers and humans in order to enable machines to understand, interpret, and generate human language. Concerning the legal domain, NLP techniques can be utilized for various tasks such as *Document analysis* [1], *Legal Chatbots* [2], *Contract Management* [3], etc. In *Case Summarization*, for instance, NLP techniques are exploited to generate concise summaries of legal cases in order to capture the most essential details and allow even those who are not experts to understand legal concepts [4].

Another promising application is *Predictive Analysis*. It consists of predicting the outcomes of legal cases by analyzing patterns (in past cases) and identifying relevant factors that may impact the final decision. An automated system capable of assisting a judge by giving an accurate prediction of the outcome of a legal case can speed

up the entire judicial process. Many highly populated countries, characterized by a huge amount of pending backlog of court cases and the unavailability of competent judges can benefit from this [5]. However, the mere prediction of a legal case is not sufficient if not accompanied by the corresponding explanation.

In this context, this thesis endeavors to tackle the task of *Court Judgment Prediction and Explanation* (CJPE), aiming to predict the outcome of a legal case and provide the corresponding explanation given the case description.

This undertaking is far more challenging than standard text-classification tasks since there are some intrinsic factors of the legal domain to consider:

1. *Complexity*: the legal domain is complex, with lots of peculiar terms that require specific skills to be navigated. These terms are often very technical and it is required a certain knowledge to understand them properly. Unfortunately, common Language Models (LM) face limitations in this context. These models are not familiar with these technical terms since they are typically trained on general language patterns and do not have the specific knowledge that is essential to understand and interpret legal concepts. Consequently, it becomes evident that the constraints of existing language models impede the effective processing of legal language [5] [6].

2. *Document length*: Legal documents, such as judgments and court decisions, are typically quite long, verbose, and noisy, since they may include a wide range of potential scenarios and contingencies. It is really challenging to find a simple method to extract and directly access the facts and arguments. Therefore, it is important to consider also the limitations of LM, in terms of the amount of data they can process when working with legal documents.

3. *"Understanding"* and *"applying"*: giving a proper explanation for a prediction requires understanding the facts, following the arguments, and applying legal rules, and principles to arrive at the final decision [5]. This can be a problem, especially in those cases in which a model is trained on legal documents characterized by different subjective interpretations of the law by their respective authors [7].

4. *Metrics of measure*: unlike traditional metrics (accuracy, precision, recall, etc.) that indicate how well a ML model performs when making predictions, assessing the quality of generated explanations is very challenging [8]. Legal cases often involve intricate details, and subjective interpretations of the law that make difficult the selection of suitable metrics to evaluate the performance of an explanability model [9].

For these reasons, the goal of this thesis is to provide a system capable of giving not only an accurate prediction of the outcome of a court case but also how it

arrives at that decision. It contributes to implementing an automated system that addresses the following tasks:

- *Court Judgment Prediction:* it consists of predicting the final decision of the court through the use of Transformer models. The analysis demonstrated that a legal pre-training enables achieving higher performance.
- *Court Judgment Prediction and Explanation:* it consists of providing the corresponding explanation of the prediction exploiting the attention mechanism and occlusions method. These approaches have contributed to reducing the disparity between the explanations generated by the machine and those produced by humans.

The chapters are structured as follows:

- The first chapter introduces the main objective of this work, giving an introduction to the problem, describing the legal context, and highlighting the limitations of NLP techniques in this domain.
- The second chapter revises some basic concepts of Natural Language Processing methods.
- The third chapter introduces the task descriptions.
- The fourth chapter gives an overview of the dataset used during the experimental activity.
- The fifth chapter includes a description of the existing works that are more related to this thesis. Some of the methods illustrated in this chapter will be further investigated in the experiments.
- The sixth chapter is dedicated to describing the methods proposed.
- The seventh chapter is dedicated to detailing the experiments conducted.
- The concluding chapter offers a summary of the study and some future directions.

Chapter 2

Natural Language Processing

This chapter introduces some basic concepts of Natural Language Processing to provide fundamental knowledge needed for comprehending the underlying principles behind the methods used in this thesis.

2.1 Fundamentals

Natural language processing (NLP) investigates the use of computational methods to process or to understand human (i.e., natural) languages to perform useful tasks. It combines computational linguistics, computing science, cognitive science, and artificial intelligence in an interdisciplinary field [10].

Due to the explosive growth of textual data available NLP has become important in recent years, especially with the help of machine learning and deep learning techniques it is now possible to solve a wide range of problems, such as sentiment analysis, machine translation, text summarization, etc.

In this sense, it is crucial to understand the context in which the language is used. But this step involves multiple challenges. Firstly, in many languages a word can be used in multiple senses, so it is important to eliminate the ambiguity of all such words. For this reason, Word-sense disambiguation is an ongoing research area in NLP whose goal is to eliminate the ambiguity of all such words so that their usage in a particular document can be detected [10]. Second, the understanding task involves documents from different domains because each domain carries a certain property that natural language understanding models should learn to capture. Third, many words could be used as proxies for other concepts. For instance, a

common legal metaphor used in court cases is *"The smoking gun"*. This metaphor is commonly used in criminal cases to refer to evidence that is conclusive and irrefutable so its meaning must be captured by terms that are not "lexically" related to the lexical items in the metaphor [11].

Nevertheless, a NLP algorithm can be considered successful if it accurately achieves its task, regardless of whether it has any explicit knowledge of the underlying linguistic structure or concepts involved.

2.1.1 Terminology

In this section, are listed some key definitions that can be found throughout the entire work.

- *Character:* is the smallest text unit. It comprises letter, digit, space, and special character.
- *N-gram:* is a contiguous sequence of N textual units, where units can be words, letters, and phonemes.
- *Token:* is the fundamental unit of NLP, it is composed of a sequence of characters used as input. Tokens can be words, symbols, or numbers.
- *Lemma:* is the canonical form of a word, chosen from a set of candidate forms in a dictionary.
- *Sentence:* is a text snippet separated by punctuation (e.g., full stop, question mark, exclamation mark).
- *Paragraph:* is a unit of text that is composed of multiple sentences.
- *Section:* is a unit of text that is composed of multiple paragraphs.
- *Corpus:* can be considered as the dataset for analysis in NLP. It consists of large and structured text that represents the input.

2.1.2 Text Representation

In the field of NLP, the representation of the data is crucial. With the recent growth of the availability of text in a digital form, to be capable of transforming human language into a representation that a machine can understand is essential. The text representation in NLP consists of the conversion of words, sentences, or documents into numbers or vectors so that they can be analyzed and processed by Machine Learning (ML) algorithms.

Bag-of-Words

Bag-of-Words (BOW) is a text representation approach widely used in NLP. The concept is quite simple but useful. BOW is a model that represents the text as a set (or "bag") of words, ignoring the order. It counts how many times a word appears in the document, or collection of documents, without taking into account the sequence of the words and their positions in the text. The entire process is divided into the following steps:

1. **Tokenization:** it involves dividing the text into words or "tokens".
2. **Vocabulary creation:** a vocabulary is created containing all the distinct words that appear in the documents.
3. **BOW vector:** for each document (or sentence), a numeric vector is created with size equal to the number of words in the vocabulary. Each position of the vector represents a word in the vocabulary, and the value in each position indicates the number of occurrences of the word in the document.

The BOW model has some limitations:

- The order of the words is ignored, but it takes into account only their frequencies. Then, the information of the context is lost, and sentences with different meanings but with the same words have the same BOW representation.
- BOW vectors can be high dimensional and can bring sparse representations in which the majority of the values that compose the vectors are equal to zero, causing an increase in term of computational cost.

Term Frequency-Inverse Document Frequency

TF-IDF (Term Frequency-Inverse Document Frequency) representation is a technique used to evaluate the importance of the words in a document or collection of documents. This approach can be divided into two parts:

- **TF (Term-Frequency):** the first component is the frequency of the words, which measures how many times a word appears in the document. This value is computed by dividing the number of times that a specific word appears in the document by the total number of words in the document. It can be computed using the following formula:

$$tf_{i,j} = \frac{n_{i,j}}{d_j} \tag{2.1}$$

where $n_{i,j}$ is the number of occurrences of the word i in the document j, and d_j is the total number of words in the document j.

- **IDF (Inverse Document Frequency):** the second component measures the importance of the word in whole document corpus. It can be expressed as the logarithm of the ratio between the total number of documents in the corpus and the number of documents that contain the word i. Its role is to value rare words that appear in a few documents and, at the same, penalize common words that appear in many documents of the corpus [12]. Mathematically, it can be expressed as:

$$idf_i = log_{10} \frac{|D|}{\{d : i \in d\}} \tag{2.2}$$

  where $|D|$ is the number of documents in the corpus.

Then, TF-IDF is calculated as:

$$tf_{i,j} * idf_i \tag{2.3}$$

2.1.3 Word Embedding

Different than traditional text representation, Word embedding tries to capture semantic relationships and contextual information. In essence, Word embedding consists of representing a word by a vector in a continuous vector space where words with similar meanings are closer.

Word2Vec

Word2Vec is a pioneering approach, proposed by *Mikolov et al.*[13] in 2013, to learn word embedding. The idea is to train a neural network in which given a word (denoted as *target word*) in a dictionary, a sliding window is built in order to slide over the text, collect samples for training, and make predictions. The goal is to predict the surrounding words of the target word. The sliding window is used to define contextual positive examples (self-supervision). Its size determines how many words before and after a given word are included as context words. The authors have also addressed the complementary task which consists on training a neural network to predict the target word given the context words. For this reason, they proposed two architectures: the *Continuous Skip-Gram model* and the *Continuous Bag-of-Word (CBOW)* model. The structure of these models is shown in Figure 2.1.

The Continuous Skip-Gram architecture consists of predicting the surrounding words given the target word. The target word is given as input to a log-linear classifier with a continuous projection layer [13], and the goal is to predict the surrounding words within a certain range of the input word.

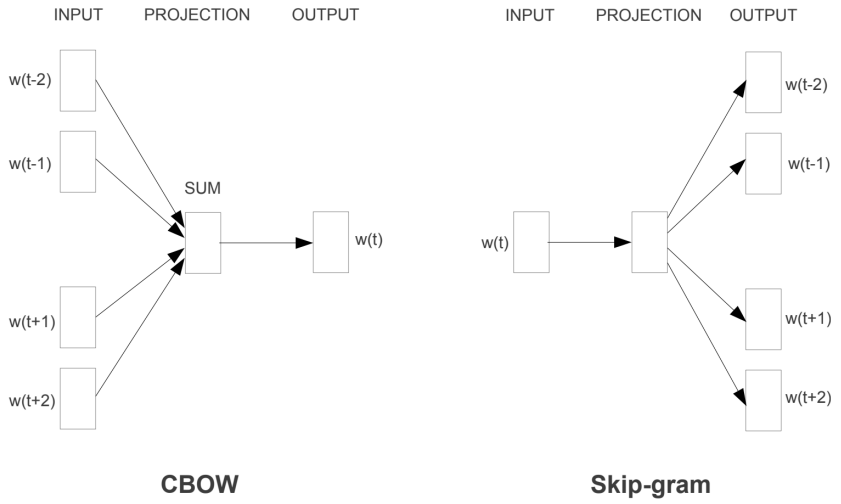


Figure 2.1: A framework for learning paragraph vector [14].

Given $w_1, w_2, ..., w_T$ the sequence of training words of cardinality T, the objective function of the Continuous Skip-Gram model is to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} logp(w_{t+j}|w_t) \tag{2.4}$$

where c is the size of the training context.

In the Continuous Bag-of-Word model, instead, the goal is to predict the target word given the surrounding words. In this case, the input layer is constituted by N words which are encoded to be passed to the projection layer that is applied to all the words. Here, a hidden vector is created, element-wise averaged, and passed to the output layer. Finally, the output layer is responsible for generating the probability distribution across the vocabulary.

Given $w_1, w_2, ..., w_T$ the sequence of training words of cardinality T, the objective function of the CBOW is to maximize the average log probability:

$$logP(w_c|w_{c-m}, ..., w_{c-1}, w_{c+1}, ..., w_{c+m}) \tag{2.5}$$

where m is the size of the training context.

These architectures have some drawbacks. Firstly, during training, only the weights corresponding to the target word might get a significant update while the weights related to the non-target words might receive only a small change or no change at all. Secondly, the calculation of the final probabilities using the softmax function is highly inefficient because the computational cost is proportional to the dictionary size.

Two further Word2Vec optimizations have been proposed to mitigate these problems:

- **Negative Sampling:** rather than attempting to estimate the probability of being a context word for every word in the vocabulary, the model tries to predict whether certain words from the training samples are included in the context or not. While Skip-Gram requires updating a large number of weights, in Negative Sampling the positive words are still updated but only a randomly selected small number of negative words will be updated [15].

- **Frequent Word Subsampling:** with this technique, the number of training instances involving highly frequent words is reduced. In this way, the overhead of updating for each occurrence of a highly common word is avoided and applied only in a subset of instances [15].

2.1.4 Sentence Embedding

For some NLP application is necessary the comprehension of entire sentences. Sentence embedders have the capability of providing contextualized embedding for words taking into account that the meaning of a word can change according the broader context of the entire sentence. Basically, these methods use all the words in a sentence to capture the context and then then embedding of each word is performed. Then, by the combination of all the word embeddings into a singe vector, the sentence embedding is obtained.

Doc2Vec

Doc2Vec, also called *Paragraph Vector* is an unsupervised algorithm designed to extract fixed-length feature representations from texts of varying lengths, such as sentences, paragraphs, and entire documents. Introduced by *Mikolov* and *Le* in 2014 [16], this algorithm creates a dense vector to represent each document, and this vector is trained to predict words in the document. Its main goal is to overcome the weaknesses of bag-of-words models.

Doc2Vec has several configurations, but the most used are *Distributed Memory Model of Paragraph Vectors* (PV-DM) and *Distributed Bag of Words of Paragraph Vector* (PV-DBOW).

In PV-DM the paragraph token can be considered as an additional word, functioning as a memory that remembers information not present in the current context or related to the paragraph's topic. Figure 2.2 shows a PV-DM framework in which the extra paragraph token is transformed into a vector using a matrix *D*. This resulting vector, when combined with a three-word context is used to predict the fourth word[14]. Furthermore, it is clearly visible in the figure 2.2 that in the

PV-DM that the context window includes only the preceding words since the target word is the next word. In contrast, in the Word2Vec architecture, the context window is composed of both preceding and subsequent words, with the target word positioned in the middle [13].

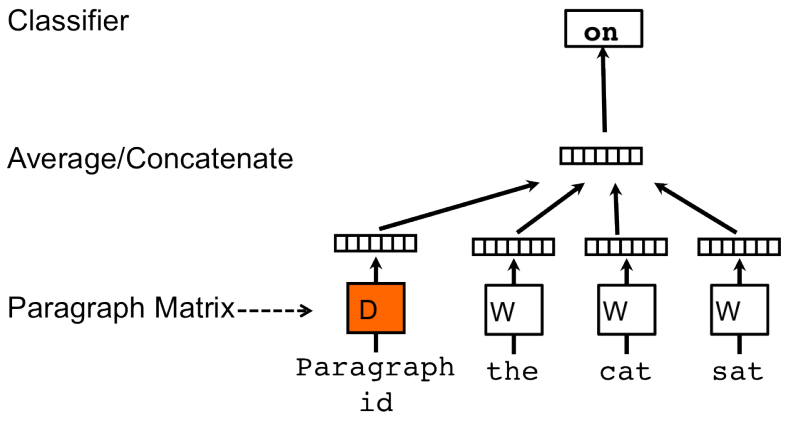


Figure 2.2: A framework for learning paragraph vector [14].

In PV-DBOW, instead, the context words in the input are ignored but this requires the model to predict words selected randomly from the paragraph in the generated output. As shown in Figure 2.3, the paragraph vector is trained to predict the words in a small window.

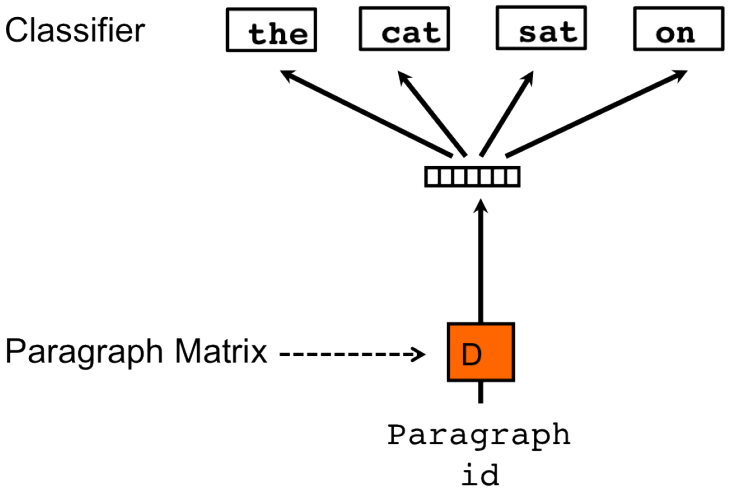


Figure 2.3: PV-DBOW version of paragraph vectors [14].

Sent2Vec

Sent2Vec is an unsupervised model, introduced by *Pagliardini et al.* in 2018 [17], designed to generate sentence embeddings by combining word vectors and n-gram embeddings. This method allows training both the composition and embedding vectors simultaneously. It offers efficiency and scalability since the computational complexity of the embeddings is equal to *O(1)* vector operations per word, whether during training or inference of the sentence embeddings [17].

The architecture of Sent2Vec consists of two main components: the encoder and the pooling layer. Typically, the encoder is implemented as a neural network, and it takes a sentence as input, producing a sequence of hidden states. Then, the role of the pooling layer is to aggregate these hidden states into a single vector representation.

The encoder in Sent2Vec is based on the Long Short-Term Memory (LSTM) architecture, which is a type of recurrent neural network (RNN). The LSTM has the ability to capture long-term dependencies in sequences, which is important for generating accurate sentence embeddings. Basically, it takes a sequence of word embeddings as input and produces a sequence of hidden states, that represent the contextual meaning of the sentence.

The pooling layer in Sent2Vec is based on a hierarchical approach that involves two levels of pooling; word-level pooling and sentence-levels pooling. At the word level, the hidden states generated by the encoder are combined using a max-pooling operation, which extracts the most significant features from each hidden state. At the sentence level, the pooled word embeddings from the word-level pooling stage, are combined using another max-pooling operation. This process generates a fixed-length vector representation that includes the contextual meaning of the sentence.

Finally, the resulting sentence embeddings can be used as input to various natural language processing tasks, such as sentiment analysis, text classification, and machine translation. The primary advantage of Sent2Vec w.r.t. other sentence embedding models is the capability of capturing the meaning of a sentence even when it contains out-of-vocabulary (OOV) words. The reason for this is that the model is trained on an extensive text corpus and can learn how to generalize new words based on their context.

2.2 Transformer Models

2.2.1 Attention Mechanism

The attention mechanism was proposed by *Bahdanau et al.* [18] in 2014. It has been introduced to mitigate one potential limitation of the Encoder-Decoder method

where it is required for a neural network to compress all essential information from a source sentence into a vector with a fixed length.

The attention mechanism can be divided into three steps:

1. Alignment: The alignment model is represented by a function f, typically implemented using a feedforward neural network:

$$e_{t,i} = f(s_{t-1}, h_i) \tag{2.6}$$

   where h_i are the encoded hidden states, s_{t-1} the previous output, and $e_{t,i}$ represents the score indicating the degree of alignment between elements of the input sequence and the current output position t.

2. Weights: The weights, denoted as $\alpha_{t,i}$, are derived by applying a Softmax operation to the alignment scores calculated previously:

$$\alpha_{t,i} = softmax(e_{t,i}) \tag{2.7}$$

3. Context Vector: A unique context vector, denoted as c_t, is given to the decoder at each time step. This context vector is computed as a weighted sum of all encoded hidden states:

$$c_t = \sum_t \alpha_{t,i} h_i \tag{2.8}$$

Initially, this method was introduced for "sequence-to-sequence" tasks where there is a sequence of input (words in a sentence), and the model needs to focus on specific parts of that input during the generation of an output.

This concept can be reformulated in cases where the input may be organized differently from a simple linear sequence. This adaptation makes use of three elements, defined as queries Q, keys K, and values V. The entire procedure is similar to the previous one and can be explained into the following steps:

1. We can think of the vector s_{t-1} as a query q that is executed against a database of keys to compute a value, where the keys are vectors and h_i are the values. This comparison is achieved through the dot product operation between the specific query q and each key vector:

$$e_{q,k_i} = q * K_i \tag{2.9}$$

2. The resulting scores are given to a softmax operation to obtain the weights:

$$\alpha_{q,K_i} = softmax(e_{q,k_i}) \tag{2.10}$$

3. The generalized attention is subsequently calculated by performing a weighted sum of the value vectors, denoted as v_{k_i}, with each value vector paired with its respective key:

$$attention(q, K, V) = \sum_i \alpha_{q,k_i} v_{k_i} \tag{2.11}$$

2.2.2 Transformers Architecture

Transformers, introduced in 2017 by *Vaswani et al.* [19], has revolutionized the field of NLP and Computer Vision. Before Transformers, the state-of-the-art solutions in NLP relied heavily on RNNs such as LSTM and Gated Recurrent Units (GRUs). However, RNNs have a sequential nature, which makes it difficult to parallelize during training.

The Transformers architecture uses an encoder-decoder architecture based on self-attention. This allows non-sequential processing and parallelization, which significantly speeds up the training process.

The input sequence is first transformed into three matrices representing keys K, values V, and queries Q. To compute the matrix of the output, the authors proposed a modified Dot-Product Attention, called "*Scaled Dot-Product Attention*":

$$attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.12}$$

where d_k represents the dimension of the queries and keys.

The Transformer architecture is composed of an encoder and a decoder, each of which contains multiple layers of self-attention mechanisms. A schema of the typical structure of the transformer model is shown in Figure 2.4.

The encoder processes the input sequence iteratively, with each layer generating encodings that contain information about which parts of the inputs are relevant to each other. The output of the encoder is given as input to the decoder. The role of the decoder is to use the context information received and generates an output sequence.

Both the encoder and decoder layers have a feed-forward neural network for additional processing of the outputs and contain residual connections and layer normalization steps. The residual connections offers the advantage of avoiding layers when they do not provide any significant information. Layer normalization normalizes the output of each layer, which speeds up the training process and reduces the risk of a poor generalization of the model, the so-called "*overfitting*".

Depending on the task, it is possible to use only one side of the encoder-decoder architecture. For example, GPT-3 [20], a state-of-the-art language model for human-like text generation, only uses the decoder side. In contrast, BERT [21], a state-of-the-art model for sentence encoding, and its variants use only the encoder side.

BERT

BERT, which stands for *Bidirectional Encoder Representations from Transformers*, Bert is designed to pretrain deep bidirectional representations from unlabelled

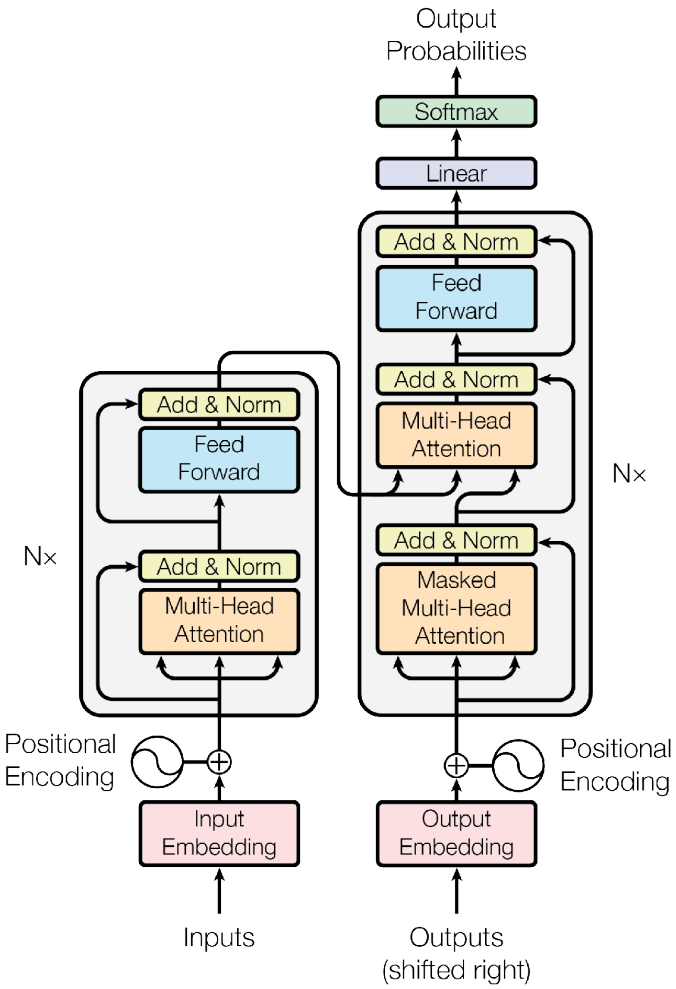


Figure 2.4: The Transformer model architecture [19].

text by taking into account both left and right context in all layers [21]. Its implementation is based on two steps: *pre-training* and *fine-tuning*.

During the pre-training phase, the BERT model is trained on two unsupervised tasks. The first task is called the *Masked Language Model* (MLM), where a percentage of input tokens are randomly masked, and the model is trained to learn these masked tokens. In particular, the training data generator chooses 15% of the input tokens at random for possible replacement. Then, 80% of these selected tokens are replaced with the [MASK] token while 10% are left unchanged and 10% are replaced by a randomly selected token in the vocabulary.

The second task called *Next Sentence Prediction* (NSP) is based on a binary classification problem. Specifically, when choosing the sentences A and B for each pretraining example, 50% of the time B is the actual next sentence that follows A,

and 50% of the time it is a random sentence from the corpus. For this reason, this task is particularly used when capturing dependencies among sentences is required.

After pre-training, the model can be fine-tuned on specific tasks such as text classification, named entity recognition and question answering. The starting point is to use the pre-trained weights of BERT, and then the model is fine-tuned on the downstream task by adjusting the weights to optimize the task-specific objective function.

It's worth noting that BERT has a token limit of 512 tokens. This limitation introduce complexity when dealing with longer documents or sentences, requiring strategies such as truncation or segmentation of the text.

Figure 2.5 shows the pre-training and fine-tuning procedures for BERT. The same architectures are used in both pre-training and fine-tuning and the same pre-trained model parameters are used to initialize models for different downstream tasks [21].

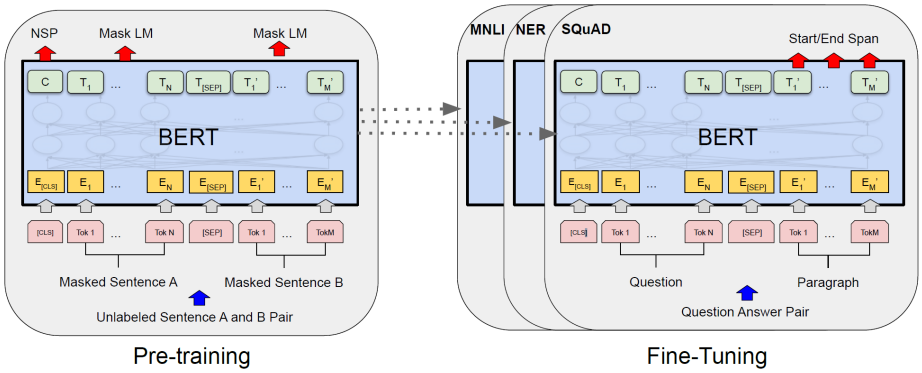


Figure 2.5: Pre-training and fine-tuning procedures for BERT [21].

RoBERTa

RoBERTa (Robustly Optimized BERT approach) is a variant of BERT introduced in 2019 by Facebook AI Research [22]. RoBERTa is based on the same transformer architecture as BERT, but it incorporates several improvements that aim to optimize its pre-training and performance on downstream tasks.

The differences with respect to BERT are many. RoBERTa was trained on a larger corpus of text data and for longer periods. Specifically, RoBERTa was trained on a combination of BookCorpus and English Wikipedia, totaling over 160GB of text data, with longer sequences and larger batches compared to BERT.

RoBERTa also uses dynamic masking, where each training example is randomly sampled and token spans are masked at random during pre-training, instead of the fixed masking scheme used in BERT. This allows RoBERTa to learn more

effectively from the vast amounts of data it is trained on, as it is forced to predict masked tokens in a more diverse range of contexts.

Another improvement in RoBERTa is the removal of the next sentence prediction objective, which was present in BERT. This objective required the model to predict whether two sentences in a given input sequence were consecutive or not. However, recent research has shown that this objective does not provide significant benefits for downstream tasks and can even harm performance in some cases. By removing this objective, RoBERTa can focus more on learning the language model itself, which can lead to better performance on downstream tasks.

RoBERTa also incorporates other optimizations, such as training the model with a larger batch size, which has been shown to improve convergence and training efficiency [22].

XLNet

XLNet was introduced in 2019 by researchers at Carnegie Mellon University and Google AI Brrain Team [23]. Same as BERT, XLNet is pre-trained on large amounts of data to learn representations of natural language, which can be fine-tuned for many different tasks. However, there are some differences with respect to BERT which are important to point out.

BERT is trained on a MLM task where a percentage of the input tokens are masked, and the model has to predict the masked tokens based on the context. The fact that BERT capture the bidirectional relationships between the input tokens has a drawback. In particular, the model can see the whole input sequence when predicting the masked tokens. Therefore, it can't learn the relationships between the tokens that are farther apart in the sequence.

In contrast, XLNet introduces a *Permutation language modeling* approach, in which the model is trained to predict the next token in a sequence given all the tokens that came before it, but in a random order. This approach allows the model to capture the dependencies between all the tokens in the sequence, not just the ones that are close to each other.

XLNet also introduces a technique called relative positional encoding, which allows the model to better understand the relationships between words based on their relative positions in the sentence. This can be particularly useful in tasks like natural language inference, where the model needs to understand the relationships between different parts of a sentence.

LED

One limitation of traditional transformer models, such as BERT [21], refers to the use of full attention mechanism, where each token in the input sequence attends to all other tokens. Basically, during the computation of the attention for a

specific token, this mechanism considers and assigns weights to all other tokens in the sequence. In this way, the model includes in the representation of a token, additional information given by the other tokens in the sequence. The main issue with this approach is the quadratic growth in the number of attention weights to be computed as the sequence length increases. This "quadratic dependency" leads to higher computational and memory requirements, especially when dealing with long sequences. The full attention mechanism relies on a matrix to capture the relationships between each token in a sequence. Each entry in this matrix represents the attention weight assigned to the relationship between two tokens in the sequence. Given an input sequence of length N, the Self-Attention operation would need to work on a matrix of dimensions $N \times N$, leading to a memory dependence of $O(N^2)$ (Figure 2.7a). To address this limitation, many approaches have been proposed over the years.

The *Longformer-Encoder-Decoder* (LED), is one of them. It is proposed in [24] as a modified Transformer architecture that incorporates a self-attention operation that scales linearly with the sequence length, providing versatility when processing long documents. The authors tried to address the quadratic growth in memory dependence of the traditional Transformers model. As shown in their example illustrated in Figure 2.6, the Longformer's memory consumption increases proportionally with the sequence length. This stands in contrast to the self-attention mechanism, which faces memory constraints when dealing with lengthy sequences (as indicated by the blue line in the graph).

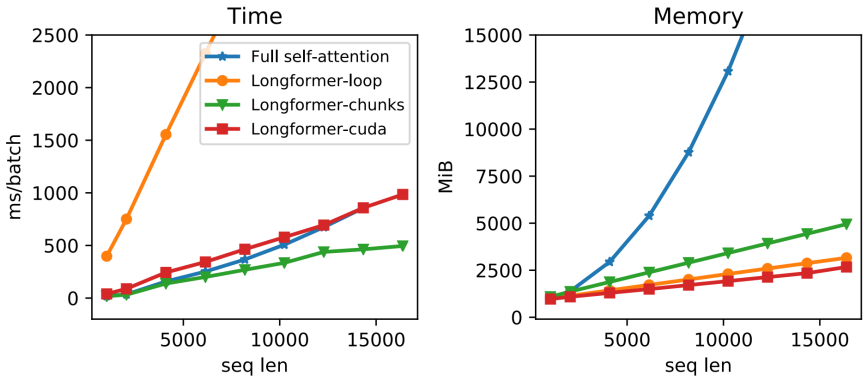
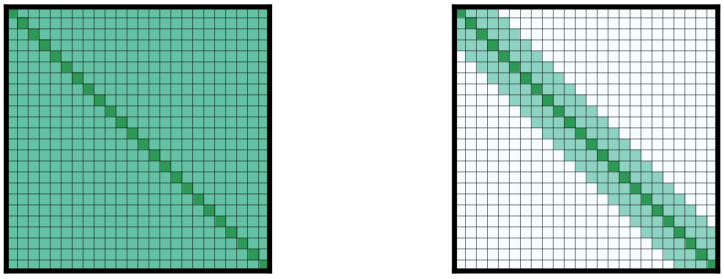


Figure 2.6: Time and Memory required for different implementations of Longformer's self-attention [24].

The self-attention mechanism in Longformer employs a fixed-size window attention that captures context over a stack of layers. If the size of the window is equal to w, each token attends to $\frac{1}{2}w$ tokens on each side (as shown in Figure 2.7b). The computational complexity for this operation is $O(n \times w)$, scaling linearly with the

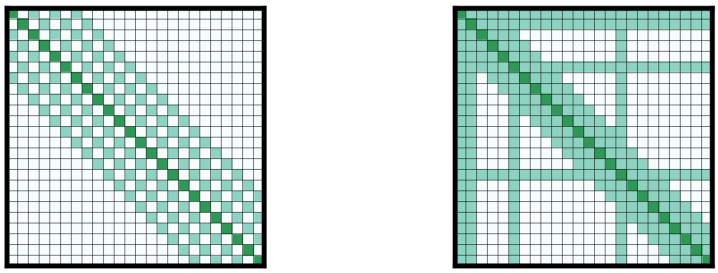
sequence length (n). If the number of layers is l the overall receptive field will be $w \times l$.



(a) Full n^2 attention (b) Sliding window attention

Figure 2.7: Comparison of the full self-attention pattern and sliding window attention [24].

Similarly, the sliding window can be "dilated" with a dilation value equal to d (as shown in Figure 2.8a). This dilation allows consecutive tokens to be skipped, making the attention matrix even sparser, and increases the receptive field to $(l \times d \times w)$ without a corresponding increase in computational cost.



(a) Dilated sliding window (b) Global+sliding window

Figure 2.8: "Dilated" sliding window attention and Global + sliding window attention. [24].

The complete attention pattern proposed by the authors is illustrated in Figure 2.8b. Every token attends to other tokens within the sliding window, while additional "global" tokens have the capacity to attend to every other token, akin to the full Self-Attention mechanism.

Concerning the encoder-decoder architecture, LED uses a combination of local and global attention. In particular, the encoder reads the input document using

local attention with a window size of 1024 tokens and global attention on the first ‘<s>‘ token[1]. Here, local attention allows to build contextual representations, while global attention is used to build full sequence representations for the prediction. The decoder, instead, uses full attention to the entire encoder and previously decoded locations. By exploiting both local and global attention, the model can capture short and long-term dependencies in the input, making it easier to process long sequences.

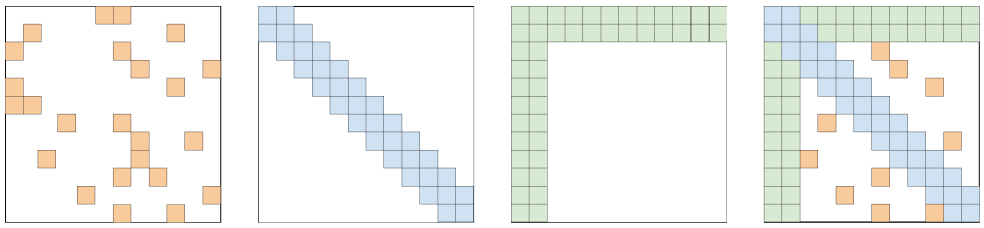
Despite its structure is very similar to BigBird [25], the authors affirm that in cases where a pre-training process or task-specific initialization is avoided, LED can slightly outperform BigBird.

BigBird

Authors in [25] introduced BigBird as a sparse-attention-based transformer model. This mechanism consists of three parts:

- *Global tokens:* they are designed to capture the most important information about the entire sequence without taking into account the position of the tokens.
- *Local Neighboring tokens:* they are designed to capture information about a specific token from its neighboring tokens within a fixed window size.
- *Random tokens:* a set of randomly selected tokens within the sequence.

By combining these three sets of tokens (Figure 2.9), BigBird can handle up to a length of 4096 tokens at a much lower computational cost than models with full attention mechanisms like BERT.



(a) Random attention (b) Window attention (c) Global Attention (d) BIGBIRD

Figure 2.9: Attention mechanism used in BigBird [25].

[1]is a token typically added in Seq2Seq models at the beginning of the target sequence to indicate the start of decoding.

2.2.3 LSG attention

LSG stands for Local Sparse Global attention. It is an architecture introduced in [26] to reduce the high computational cost caused by the full attention mechanism in the traditional transformer models. The LSG attention is based on three components:

- **Local Attention:** to improve the efficiency in terms of the computational cost of the local attention in Longformers [24], the authors were inspired by BigBird [25]. Basically, they apply a block-based process in which the sequence is split into blocks of size b_t without overlapping. Each token in each block "pays attention" to the other tokens within the same block but, at the same time, it also considers the tokens in the preceding and succeeding blocks. While Longformers are characterized by a fixed-length sliding window, in this configuration the local attention window is asymmetrical. In particular, a token connects to a maximum of $2 \times b_t - 1$ tokens to the left or right.

- **Sparse Attention:** sparse connections serve to broaden the local context by picking further tokens according to a set of rules. These tokens can be chosen either directly based on a particular metric or through a pooling strategy. Each attention head has the capability to process distinct sparse tokens autonomously. Additionally, sparse attention is dependent on a block structure, where the selection of sparse tokens occurs within each block. The authors proposed five criteria that can be used: *Head-wise strided* [27], *Head-wise block strided*, *Average pooling*, *Max norm* (default method), and *LSH Clustering* [28].

- **Global attention:** global tokens are used to improve the information flow within the model. Unlike the conventional method of selecting a subset of tokens and labeling them as global, this approach involves prepending global tokens to the sequence. These global tokens possess their own embedding matrix, representing an additional hyperparameter in the model. During the conversion to the LSG version, the initialization of the initial global token involves summing the [CLS] (or <s>) token with the first position from the positional embedding. The initialization of subsequent global tokens, instead, is performed by summing the [MASK] (or <mask>) token and other positions from the positional embedding.

Using this approach, the authors shown that LSG attention is fast and efficient. Furthermore, they also proved that various existing pretrained models can be converted to their corresponding LSG variant to handle long sequences. An open

source library[2] who is in charge of doing that is provided by the authors.

Figure 2.10 show a comparison of LSG, BigBird, and Longformer. The intuition behind these methods is to have a sliding window attention that reduce the memory complexity from $O(N^2)$ to $O(N)$, and global tokens that improve the flow of information among the tokens.

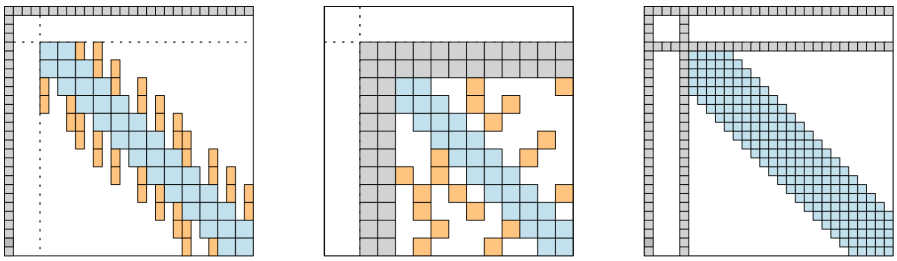


Figure 2.10: A comparison of LSG, BigBird, and Longformer attention patterns [26].

[2]https://github.com/ccdv-ai/convert_checkpoint_to_lsg

Chapter 3

Problem Formulation

3.1 Task Descriptions

Given a court case of the Supreme Court of India, the goal of this thesis is to build a model that provides an automatic prediction of the case outcome together with an explanation for the decision. This procedure can be divided into two sub-tasks: *Court Judgment Prediction* and *Court Judgment Prediction and Explanation*.

Court Judgment Prediction

The Court Judgment Prediction (CJP) task is a binary classification problem in which the objective is to predict the final decision of a court case. Formally, given a document d representing the case description, the task consists of predicting the decision $y \in \{0,1\}$ where class 0 represents the rejection of the appeal while class 1 represents acceptance.

Court Judgment Prediction and Explanation

The Court Judgment Prediction and Explanation (CJPE) task, instead, has the objective of extracting the key sentences from the case description that better justify the final decision given by the previous CJP task. Since human explanations are very difficult to obtain, manually annotated documents are not provided during training, in order to obtain a model learned to make predictions that is capable of generating explanations without being explicitly trained on them. Formally, given the predicted decision y, and the set of all the sentences $E = \{s_1, s_2, ..., s_n\}$ of the document d, the task can be expressed as:

$$E^* = \arg max_E F(E, y) \tag{3.1}$$

where F is the objective function that measures the quality of the explanations, and $E^* \in E$ is the set of important sentences that maximizes this objective. The cardinality of E^* is equal to m with $m \leq n$.

Chapter 4

Dataset Overview

This chapter is dedicated to providing an overview of the dataset used during the experiments.

4.1 Dataset

All the experiments have been conducted using the ILDC (Indian Legal Documents Corpus) introduced in [5]. It is composed of a large set of Indian Supreme Court cases, expressed in the English language and annotated with original court decisions. As stated in [5], in a Supreme Court of India (SCI) case, the judge decides whether to accept or reject the appeal or petition by weighing the arguments, relevant statutes, precedents, and the facts of the case.

The documents in ILDC are cleaned by removing metadata information such as case numbers, names of judges, dates, etc. In addition, the final section of each document that contained the final decision was removed so that it could be predicted. The labels are binary, where '0' represents rejection and '1' represents acceptance.
The ILDC corpus includes:

1. $ILDC_{single}$: contains documents with a single petition (and therefore a single decision) or multiple petitions with identical decisions for all those petitions. Its training set is composed of 5082 documents. Figure 4.1 shows the class distribution of $ILDC_{single}$ in which the percentage of accepted cases (*class* 1) is 38.08%.

2. $ILDC_{multi}$: is a superset of $ILDC_{single}$ and includes multiple appeals resulting in different decisions. Its training set is composed of 32305 documents. Figure 4.2 shows the class distribution of $ILDC_{multi}$ in which the percentage of accepted cases (*class* 1) is 41.43%.

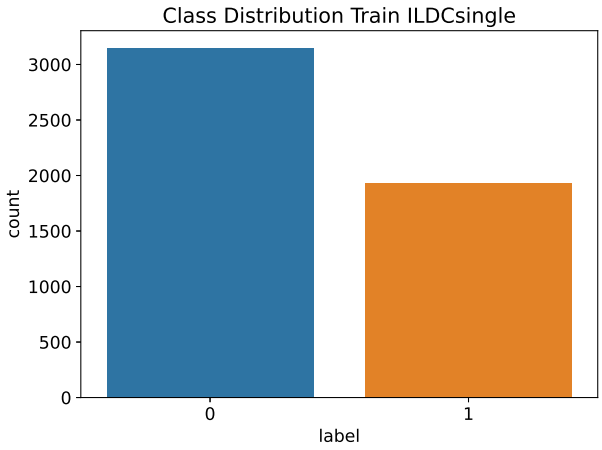


Figure 4.1: Class Distribution in the Training Set of $ILDC_{single}$.

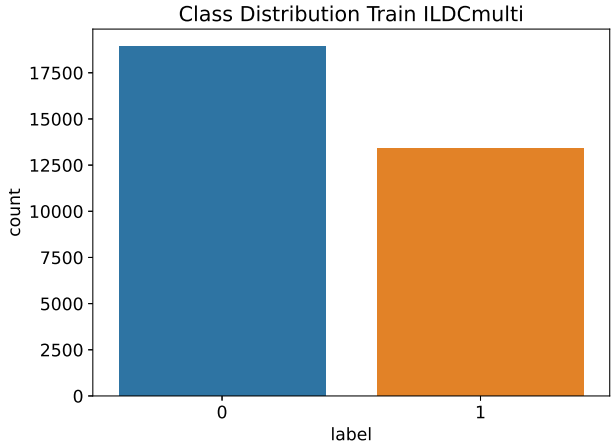


Figure 4.2: Class Distribution in the Training Set of $ILDC_{multi}$.

The Validation and Test sets are shared by both $ILDC_{single}$ and $ILDC_{multi}$. The class distribution is almost perfectly balanced with the percentage of the accepted class of 50% in the Validation set and 50.23% in the Test set.

In addition, for a collection of 56 documents in the test set, annotations of decision explanations are obtained from 5 legal experts. This subset constitutes the $ILDC_{expert}$. The role of the legal experts is to predict the judgment and mark the sentences that they think could explain their final decision. In particular, the sentence marking is based on a ranking in which sentences that immediately contribute to the decision are assigned to higher ranks ($Rank1$ is the highest). The total number of ranks is 10 and, naturally, only a small portion of the sentences

have been ranked.

The level of accuracy reached by the experts in predicting the outcomes of the legal cases is 94%.

Table 4.1 shows in detail the statistics of the ILDC corpus. Notably, the "Average Tokens" column indicates a relatively lengthy mean for the number of tokens, especially considering that these tokens represent the input data. Additionally, the $ILDC_{Expert}$ stands out for its smaller size, indicating how obtaining human-annotated explanations is a time-consuming task that requires considerable manual effort.

An example of document contained in $ILDC_{multi}$ is reported below:

'M. JOSEPH, J. The appeal is directed against the Order of the High Court setting aside the Order passed by the Magistrate allowing the application filed by the appellant to discharge him. The charge-sheet came to be filed on the basis of a FIR dated 01.10.2011. The appellant was Director of Mines and Geology in the State of Karnataka at the relevant time. Signature Not Verified There was a partnership firm by the name M s Associated Digitally signed by ANITA MALHOTRA Date 2020.01.07 175329 IST Reason Mineral Company AMC, for short . The offences are alleged to revolve around the affairs of the said firm. First accused is the husband of the second accused. They became partners of the firm AMC in 2009. Appellant was arrayed as the third accused. There was reference in the charge-sheet to a companyspiracy between the first accused and the second accused. It is alleged, inter alia, that they obtained an undated letter from one Shri K.M. Vishwanath, the Ex-Partner, which is after his retirement with effect from 01.08.2009 from the firm, which was addressed to the appellant, seeking directions to the Deputy Director of Mines and Geology, Hospet in Karnataka to issue the Mineral Dispatch Permit MDP for short to the new partners, viz., the first accused and the second accused. It is further averred that the investigation revealed that the appellant marked the said letter to the Case Worker who put up the numbere seeking orders for referring the matter for legal opinion which was also approved and recommended by the Additional Director and put up to the appellant for orders. Appellant is alleged to have acted in pursuance to the criminal companyspiracy and abused his official position with a dishonest and fraudulent intention to cheat the Government of Karnataka and knowingly made a false numbere in the file that he had discussed this matter with the Deputy Director Legal and directed Deputy Director, Mines and Geology, Hospet for issue of MDPs to the new partners, viz., the first accused and the second accused by violating Mines and Minerals Development and Regulation Act, 1957 hereinafter referred to as the Act, for short and Mineral Concession Rules, 1960 hereinafter referred to as the Rules, for short . There are various allegations regarding other accused. As far as appellant is companycerned, it is alleged further in the charge-sheet that the acts of the accused,

seven in number, including the third accused appellant , companystitutes criminal offences punishable under Sections 120B, 420, 379, 409, 447, 468, 471, 477A of the Indian Penal Code, 1860 hereinafter referred to as the IPC, for short and Sections 13 2 and 13 1 c and 13 1 d of the Prevention of Corruption Act, 1988 [...]'

Corpus	**Number of Documents**			**Average Tokens**	**% Accepted Class (1)**	**% Rejected Class (0)**
	Train	**Validation**	**Test**			
Single	5082	994	1517	3884	38.08%	61.92%
Multi	32305			3231	41.43%	58.57%
Expert	56			2894	51.78%	48.22%

Table 4.1: Statistics of ILDC corpus. The "Average Tokens" column refers to the length of input sequence.

Chapter 5

Related Works

This chapter describes some of the existing works that are more closely related to this thesis, initially focusing on a broader research area before delving into the Court Judgment Prediction and Explanation task. The goal is to understand the basis of the methods used, highlight the evolution in this research area, identify limitations or possible solutions.

5.1 Legal Document Understanding

In the field of Natural Language Processing (NLP) for legal applications, it's crucial to understand and classify legal documents effectively. This section acts as a guide, highlighting recent research aimed at tackling the complexities found in legal texts. This wider exploration includes models that are not directly focused on prediction and explanation tasks, but play a crucial role in the broader objective of comprehending and categorizing legal documents.

LegalBERT is one of them. It has been proposed in [29] to investigate the use of BERT models in the legal domain. The authors released a family of BERT models for the legal domain based on two strategies:

- **LegalBERT-FP:** where "FP" stands for "further pre-training". It consists of taking a pre-trained BERT model and further pre-training it on a domain-specific corpora.
- **LegalBERT-SC:** where "SC" indicates "pre-train from scratch". It consists of training a BERT model from scratch on domain-specific corpus with a new vocabulary of sub-word units. Naturally, this approach is more time-consuming with respect to the previous one but it could lead to better performance.

As expected both models outperformed the generic BERT-BASE model on multiple legal NLP tasks. For the pre-train process, the authors collected 12 gigabytes of

English legal documents including legislation, contracts, and court cases scraped from public resources. This thesis explored the usage of the largest pre-training configuration of LegalBERT-SC. The corresponding training corpus is shown in Table 5.1 and comprises 354,624 US/EU legal documents.

Corpus	Number of documents
EU legislation	61,826
UK legislation	19,867
European Court of Justice cases	19,867
European Court of Human Rights cases	12,554
US court cases	164,141
US contracts	76,366

Table 5.1: Training corpus of LegalBERT [29]

Authors in [30], instead, proposed CaseLawBERT, another variant of the BERT model. To develop CaseLawBERT, they conducted pre-training from scratch for 2 million steps, utilizing the Harvard Law case corpus[1] from 1965 to 2021. Additionally, it features a domain-specific legal vocabulary created using SentencePiece[2] on a subset of, approximately, 13 million sentences from the pretraining corpus, with a fixed number of tokens of 32,000. This means that the model can only recognize and generate words that are present in this vocabulary.

LegalLSGBERT, instead, is an adaptation of LegalBERT [29] that exploits the Local Sparse Global attention introduced in Section 2.2.3. It uses the same pretraining corpus (Table 5.1), the same number of parameters, and the same tokenizer.

Similarly, LegalLED is a variation of LED designed for legal document summarization. The only difference is that this model is trained on the *sec-litigation-releases*[3] dataset. This dataset consists of 2700 litigation releases concerning civil actions brought by the US Commission in federal court.

5.2 Court Judgment Prediction

The evolution of CJP research can be split through three distinct phases: early statistical models, machine learning-based approaches, and more recently, the introduction of neural models.

[1] https://case.law/

[2] A popular open-source tool for sub-word text segmentation [31]

[3] https://www.sec.gov/litigation/litreleases

Early Statistical Models Some of the earliest research in this field explored mathematical models and statistical correlations. The author in [32] introduced one of the first statistical models to predict decisions of the Supreme Court of the US, already in 1957. In his work, he identified factual elements that could influence the Court's decisions. A similar work was presented in [33] in which the author explored three methods (correlation, regression, and discriminant analysis) to measure the statistical relationships between some pre-determined case factors (evidence) and the actual Supreme Court decisions. In [34], instead, the authors used historical voting records to explore how the ideologies of "median justices" can influence US Supreme Court decisions, finding out that the median justice could significantly guide the predictions.

However, while being excellent research for those years, their application in the real world is practically impossible considering the extensive hand-engineering that was required and the reduced size of the datasets used (100 documents in [35] and only 20 in [33]). In relation to this, an extensive feature engineering was required.

Machine learning-based approaches With the advances in ML, subsequent research has focused on obtaining a more efficient feature representation of legal text. Authors in [36] tried to select surface-level features (keywords and bi-grams) from legal documents in Chinese language. Then, they combined them with latent semantic features obtained through PCA and LDA to predict the accusations. In [37] and [38], the authors used Support Vector Machines (SVM) in combination with BOW features to predict European Court of Human Rights' decisions. However, even these methods require an extensive hand-engineering of the features including the composition of the BOW vocabulary. Furthermore, the BOW representation of the words, as already discussed in Section 2.1.2, has the main limitation of not taking into account the order of the words and the general semantics of the document. For these reasons, the neural models have overcome these approaches.

Neural Models Recently, neural network models have brought significant improvements in the Legal Judgment Prediction Task. Instead of leveraging manual feature engineering, these models are capable of extracting predictive features directly from the case descriptions.

Authors in [39] proposed an attention-based neural network method that jointly models the charge prediction and relevant article extraction in a unified framework. Despite they asserted that their model is capable of generalizing well since it obtained good performance with news articles not related to the legal domain, they notice that the model cannot explicitly handle multi-defendant cases and there is still a wide gap between their proposed model and the upper bound improvement that relevant articles can achieve.

In [40], instead, the authors proposed a topological multi-task learning framework, called "*TopJudge*". Their framework incorporates structural dependencies in the form of a Directed Acyclic Graph (DAG). However, their method under-perform in cases of imbalance of the category labels since the model fails to distinguish between cases with no penalty and those with a short term of imprisonment.

The works in [41] and [42] focused on the common challenge of the "confusing charge pairs". The main limitation in this context is the lack of training data. Authors in [41] to overcomes this by utilizing attribute-based representation learning, while in [42] the authors implemented a graph neural network with an attention mechanism that learns differences between confusing law articles and extracts significant discriminative features from fact descriptions.

Authors in [43], tried to overcome the length limitation of BERT [21]. They proposed a hierarchical version of BERT in which BERT-BASE generates fact embeddings by reading the words within each fact of a case. However, their method tends to under-perform in cases of "few-shots labels".

Finally, the author in [5], introduced the CJPE (*Court Judgment Prediction and Explanation*) task. Their work is close to this thesis since they tried to construct an automated system that provides predictions of legal cases in ILDC, as well as the corresponding explanations. They experimented with a set of Transformers together with their corresponding Hierarchical version (Figure 5.1). Due to the limitation in terms of the amount of tokens that these Transformers can process (512 tokens), they experimented with different sections of the documents. Their results show that the Transformers models outperformed Classical models when the training was done by using the last 512 tokens of the documents in $ILDC_{multi}$. However, the main limitation of their approach is having used only generic transformers models. Authors in [44] and [45] have demonstrated that models like BERT tend to under-perform in specialized domains such as biomedical or specialized text. For this reason, domain-specific transformers pre-trained on the legal domain could improve the performance. Furthermore, the battery of transformers exploited were capable to handle only 512 tokens.

5.3 Explainable Artificial Intelligence

Explainable Artificial Intelligence (XAI) refers to the capability of a system to explain the decisions taken in a way that is understandable for humans. Many ML models are considered "black boxes" during the prediction process since it can be difficult to understand how they arrived at their predictions. This lack of transparency can be an issue in various domains where it is essential to understand the reasoning behind a decision. This section introduces some existing techniques proposed to address this issue that are not closely related to the legal domain.

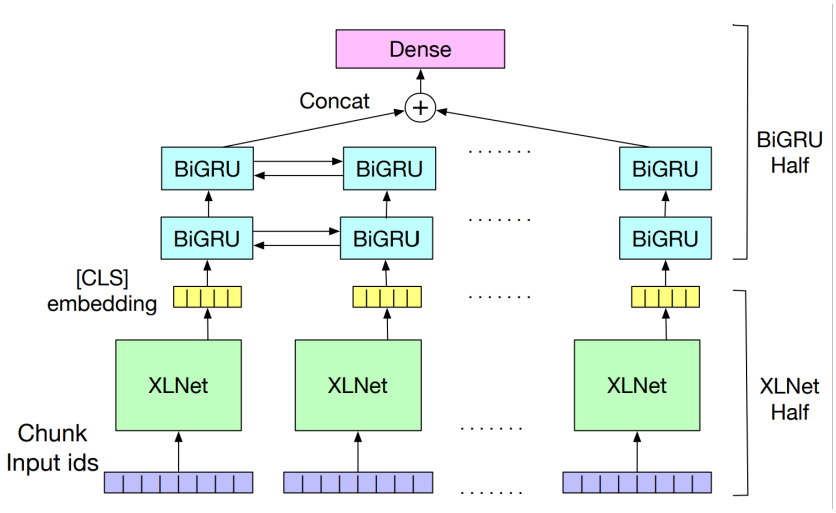


Figure 5.1: Hierarchical architecture of XLNet used in [5].

The work proposed in [8] introduces *LIME* which stands for Local Interpretable Model-agnostic Explanations. It consists of creating local and interpretable explanations for the prediction generated by a model. The explanation creation it is based on the generation of perturbed instances around a particular data point. LIME is model-agnostic, meaning that it can be applied to any machine learning model without knowledge of its internal structure. Despite being a powerful technique it suffers from some limitations. The number of perturbed instances to be generated and evaluated could make the task infeasible in the case of large datasets. It does not work well with those models that rely on temporal or sequential data. Furthermore, it may not be able to capture context information in case of noisy or incomplete data.

Authors in [46], instead, introduced SHAP (*SHapley Additive exPlanations*), a unified framework that assigns a contribution score to each feature in a prediction, indicating its impact on the output of a model. In essence, it provides a global understanding of feature importance across the entire dataset. Similar to LIME, SHAP is model-agnostic, and computing its values can be computationally expensive for large datasets or complex models. Furthermore, it assumes independence among the features, which might not hold in some cases.

5.4 Court Judgment Prediction and Explanation

In the legal domain, ensuring transparency, interpretability, and user trust is crucial. Obtaining explanations that best justify the reasons behind the legal decision-making process of an automated system is an ongoing challenge. Various

approaches have been explored to provide textual explanations, some of them are mentioned in this section.

Authors in [47] introduced a method that produces textual explanations from fact descriptions of a criminal case. The primary drawback of their approach is that the explanations are incorporated into the training data, in contrast to what is done in this thesis, where the human-annotated explanations are not included during training in order to obtain a model learned to make predictions that is capable of generating explanations without being explicitly trained on them.

The work presented in [48], introduces a two-step approach to provide explanations of charge predictions based on three components: *Extractor*, *Rewarder*, and *Classifier*. The authors affirm that their method cannot handle potential bias in the training data, resulting in a drastic drop in performance and generalization ability.

Authors in [49] proposed "*QAjudge*" a model for LJP that tries to provide interpretable judgment using reinforcement learning. This method suffers significantly when parts of the case description have been omitted.

As already mentioned in the previous section (5.2), in the CJPE task introduced in [5], the authors aim to produce explanations for the generated predictions. They explored three classes of explainability models: attention-based, model-agnostic, and attribution-based. Some of these methods were not applicable due to the extensive length of the ILDC documents, others provided explanations composed of short sentences or even just a few tokens. Taking inspiration from the works presented in [50] and [51], the authors proposed a method based on the use of the occlusions method at both levels of the hierarchy. Their results show a wide gap between the machine-made explanation and the way a legal professional would manually generate it. Furthermore, differently from what was done by the annotators, their proposed explainability models are not capable of providing a rank of the sentences extracted to compose the explanation.

Chapter 6

Methodologies

This chapter is dedicated to presenting the proposed methods for the Court Judgment Prediction and Explanation task. In particular, the first section refers to some existing approaches applied to generate the predictions, while the second section illustrates some possible solutions concerning the explanation generation task.

6.1 Court Judgment Prediction

This section makes a comparison between the transformers model applied and their corresponding hierarchical version.

Non-Hierarchical Transformer models A battery of Transformers, including both *Generic* models (RoBERTa, BigBird, LED) and *Domain-specific* models (LegalBERT, CaseLawBERT, LegalLSGBERT, LegalLED) are employed during the experiments. The initial step involves experimentation on different sections of the ILDC documents. Following the methodology outlined in [5], all the experiments focus on the last N tokens of the documents where N is equal to the maximum amount of tokens supported by each Transformer model. In this phase, both $ILDC_{single}$ and $ILDC_{multi}$ are used.

Hierarchical Transformer Models Taking inspiration from [43] and [5], Hierarchical Transformer Models are employed to further increase the performance. In this phase, all the documents contained in the $ILDC_{multi}$ are split into chunks of 512 tokens with an overlapping of 100 tokens. Using Transformer models hierarchically requires fine-tuning these models on the downstream task of classification.

Two different strategies are used:

- Considering the $ILDC_{multi}$, the last N tokens of each documents are exploited, where N is equal to the token limit of each base model (e.g., 512 for LegalBERT).
- Concerning the $ILDC_{single}$, since it is smaller with respect to the $ILDC_{multi}$, a data augmentation technique is applied. In particular, to fine-tune the transformer, each document in $ILDC_{single}$ is divided into chunks of 512 tokens with 100 tokens of overlap. Then, to each chunk is assigned the same label of the whole document.

Then, three methods are used to extract the embeddings in a 768-dimensional space [52]:

- **[CLS] token:** it stands for "classification" and can be considered as a special token used to represent the entire sequence because it contains information about the entire input.
- **Mean pooling:** it consists of taking the average of all the vectors in the sequence. Basically, given a sequence of vectors $[v_1, v_2, ..., v_n]$ the mean pooling representation is computed as:

$$\frac{1}{n}\sum_{i=1}^{n} v_i \tag{6.1}$$

  This method is often used to captures the overall information in the sequence.
- **Max pooling:** it consists of taking the element-wise maximum of all the vectors in the sequence. Given a sequence of vectors $[v_1, v_2, ..., v_n]$ the max pooling representation is computed as:

$$max(v_1, v_2, ..., v_n) \tag{6.2}$$

  This method is useful to focus on the most important features in the sequence.

Finally, the extracted representations of the chunks are used as input for a sequential model that includes two layers of standard BiGRU. Figure 6.1 shows the architecture just described, while Figure 6.2 illustrates the three pooling strategies applied after the fine-tuning process of the base model.

6.2 Court Judgment Prediction and Explanation

This section is devoted to the most challenging part of this thesis: the explanation generation task. Given the case description of an $ILDC_{expert}$ case and the corresponding predicted decision obtained with the CJP task, the goal is to predict the

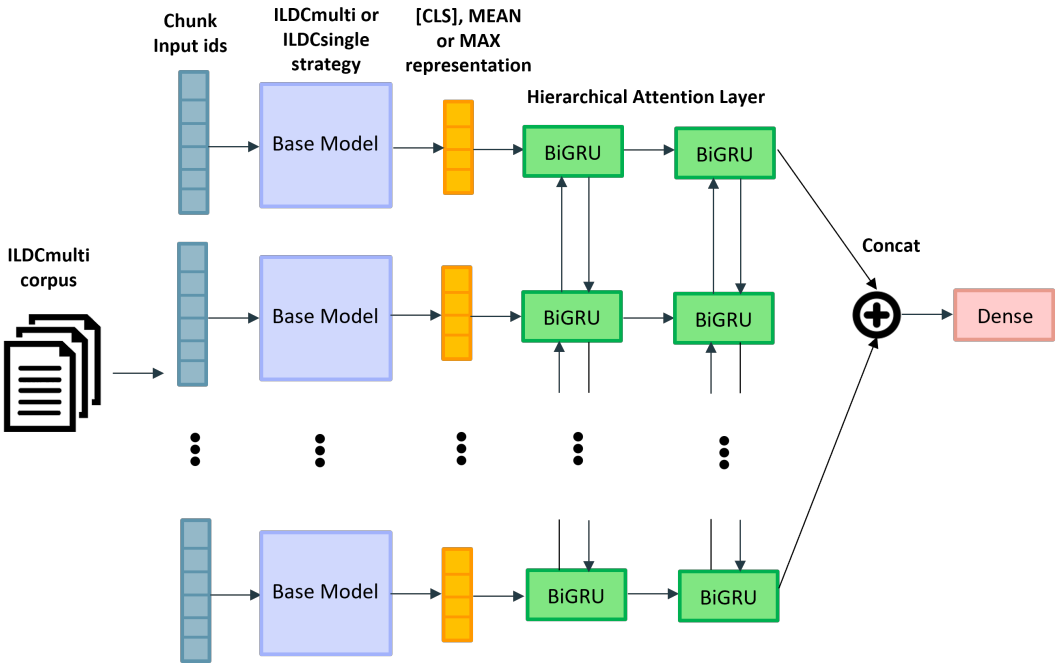


Figure 6.1: Hierarchical Transformer Models architecture similar to [5].

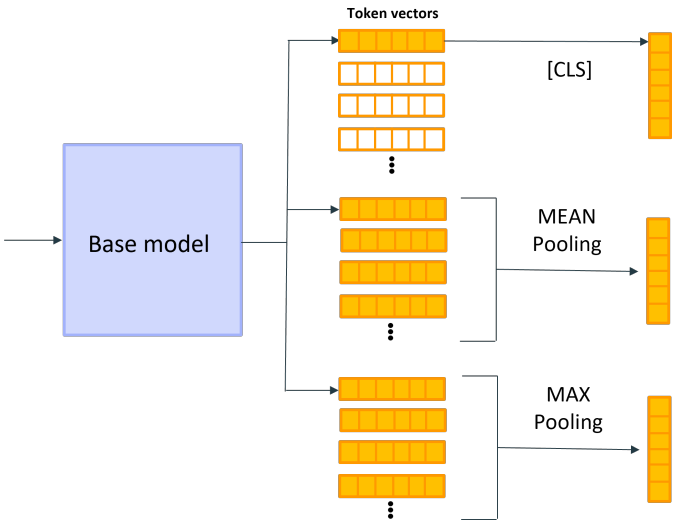


Figure 6.2: Pooling strategies applied after the Base model block in Figure 6.1.

most important sentences that better explain the decision. To accomplish this, the occlusions method and the attention mechanism are exploited to determine which segments of text better explain the predictions.

The following methods are applied using the two most performing models

obtained by the CJP task: LegalLSGBERT+BiGRU and CaseLawBERT+BiGRU (both using the [CLS] token). Also in this case, the chunks are obtained by splitting each document of the $ILDC_{expert}$ into 512 tokens with an overlap of 100.

Occlusions Method The occlusions method is implemented following the same procedure applied by the authors in [5]. Their approach can be summarized into the following steps:

1. For each document in the $ILDC_{expert}$ a chunk embedding is masked one at a time.
2. The trained BiGRU receives the masked input and using the original unmasked model it computes the "masked" probability of the input.
3. To obtain the occlusion score, this probability is compared with the original unmasked probability.
4. The occlusion scores obtained are used to filter chunks with negative scores.
5. The remaining chunks are segmented into sentences, each sentence is masked using a list of [PAD] tokens with dimension equal to the number of tokens in the sentence.
6. Similar to step 2, the logits of the chunk with the masked sentence are compared with the logits of the label corresponding to the original hierarchical model.
7. To obtain the explanation score of each sentence, the difference between these logits normalized by the sentence's length is computed.
8. The explanations are constructed by extracting the top-k sentences in each chunk.

Hierarchical Attention Mechanism To assign the attention weights a Hierarchical attention Layer on top of the BiGRU is used. The value of these weights depends on the importance of the chunk in label prediction. The attention layer is implemented taking inspiration from the work in [53]. The authors proposed a Hierarchical Attention Network (HAN) composed of two layers of attention: word-level and sentence-level. At the word-level, the model attends to different words in a sentence to construct a sentence vector. This is done using a GRU-based sequence encoder that generates a hidden state for each word in the sentence. Then, the attention layer at word-level computes a weight for each word based on its importance in the sentence representation. The sentence vector is constructed by by computing a weighted sum of the word vectors. At the sentence-level, instead,

the model attends to different sentences in a document to construct a document vector. The process is similar since this attention layer computes a weight for each sentence based on its relevance to the document representation. These weights are used to compute a weighted sum of the sentence vectors, which is the document vector. The only difference is the set of parameters used, since they operate at different levels of granularity the optimization changes to capture the most relevant features at their respective levels. To compose the explanation, the chunk with the highest attention score is selected. Then, through a detokenization function, the tokens of the chunk are joined together. The output represents the explanation of the legal case.

Revised chunk division As stated in [5], the most relevant section that influence the prediction is the end of the document. But, it is important to consider that when the chunk division of the documents starts from the beginning, the final chunk in each document typically has significantly fewer tokens than the preceding one. This can cause a potential risk of fragmenting the most relevant information into multiple chunks, potentially leading to a decline in performance.

An example of this scenario is seen in the document "*1967_359.txt*" of the $ILDC_{expert}$, as shown below:

'' the petitioner herein is a displaced person from west pakistan. after companying to india he occupied as tenant the entire first floor of the property number xvi/1588 old 1674 new situate in 35 naiwala karol bagh new delhi. the ground floor of that building was originally occupied by anumberher tenant by name hari singh. according to the petitioner hari singh vacated the portion of the building which he was occupying some time in 1956. thereafter some unauthorised persons were occupying the same. 2 28th december 1956 the petitioner applied for transfer of the building in question in his favour. the established facts are- he is a displaced person he has numberverified claim he is the lawful occupant of a portion of the premises mentioned earlier the other portions of the building are in possession of unlawful occupants and the premises in question is an acquired evacuee property which is an allottable property. all the authorities below have rejected the petitioners claim. hence he has companye up with this petition under article 226 and 227 of the companystitution. in this petition he has prayed for two reliefs namely i to quash the orders passed by the respondents by issuing a writ of certiorari and ii to issue a writ of mandamus to them requiring them to allot the premises in question to him. on the date the petitioner made his application for transfer of the property in his favour namely 28th december 1956 the two relevant rules in force were rules 26 and 31 of the displaced persons companypensation and rehabilitation rules 1955. rule 26 to the extent it is necessary for our present purpose reads as follows- where an acquired evacuee property which is an allottable property is in the sole occupation of displaced persons who does number hold a verified claim the property may be

transferred to him - the remaining portion of the rule is number relevant for our present purpose . 31. 1 where an acquired evacuee property which is an allottable property is in occupation of more than one displaced person numbere of whom holds a verified claim the property may be transferred to the displaced person who occupies the largest portion of the property or where two or more such displaced persons occupy a portion of the property which is equal in area the property may be transferred to the displaced person who has been in occupation of such portion for a longer period. the provisions of rule 26 shall apply to the transfer of acquired evacuee property under this rule in the same manner as they apply to the transfer of such property under that rule. rule 31 was abrogated on 3rd august 1968. the authorities below have rejected the claim of the petitioner on the ground that the premises in question having been occupied by more than one occupant rule 26 is number applicable to this case and further as rule 31 had been abrogated before this case was decided he companyld number take the benefit of that rule. both under rule 26 as well as under rule 31 only a discretion is given to the authorities to allot the property to a displaced person. the displaced person has number been companyferred with any right to have the property transferred to him. that being so numbermandamus can be issued to the authorities to companypel them to allot the property in favour of the petitioner. rule 26 which is still in force applies only to cases where an acquired evacuee property which is an allottable property is in the sole occupation of a displaced person. on his own showing the petitioner is number in the sole occupation of the premises in question. he is in occupation of only one floor therein. the remaining portions in the building are in occupation of unauthorised persons. it is true the word occupation found in rule 26 refers to lawful occupation. but then before a person can take then benefit of rule 26 he must be in the sole occupation of the entire building. a partial occupation of a building by him though its remaining portions are unumbercupied does number give the displaced person the benefit of rule 26. therefore the authorities were right in holding that the petitioner cannumber have the benefit of rule 26. number companying to rule 31 as mentioned earlier when the petitioners case came to be decided that rule had been abrogated. as seen earlier the petitioner has numbervested right to get the property transferred in his favour. therefore one rule 31 is abrogated the discretion companyferred on the authorities ceased to exist. they had to decide the matter before them in accordance with the law in force at that time.''

This document is divided into three chunks. The first, represented by the color red, achieved an attention score of 0.0010. The second represented by the green color, scored 0.7842, while the last one (blue) scored 0.2147. To be noted, the black-colored portion of the text represents the tokens shared by consecutive chunks.

This is a case in which the most important part of the document is contained in both the penultimate chunk and the last one. Since the penultimate chunk

contains further tokens that might be helpful for the prediction, it achieved a higher attention score.

In order to make maximum use of the width of the last chunk and avoid relevant information being divided into multiple chunks, a different corpus preprocessing is applied than the original method applied in the CJPE task [5]. In particular, the corpus is again divided into 512 tokens with 100 tokens of overlap, but in this case, the division has been made to start from the end so that the last chunk will always be full and any unfilled chunk will be the first one (since it contains less relevant information). Using this method, the totality of the documents in the test set has the last chunk as the most representative one.

Chapter 7

Experiments

This chapter is dedicated to discussing the results obtained with the proposed methods described previously. In particular, the first section shows the experimental parameter used (optimizers, learning rates, etc.), the second section describes the metrics exploited during the evaluation of the models and the third section illustrates the results obtained. Each section is divided into two subsections to make a distinction between the CJP and CJPE task. Concerning the CJP task, the experimental activity focuses on three key aspects:

1. A comparison between Generic Transformers models and Domain-specific Transformer models. The aim is to assess whether a base model pre-trained on a corpus that includes legal documents (even if not specifically coming from the Indian legal system) can yield better performance.

2. A comparison between each Transformer model with the corresponding Hierarchical version. The goal is to determine if the hierarchical structure of transformers leads to an improvement in performance.

3. An analysis concerning the calibration of the predictions. The aim is to explore whether transformers applied hierarchically can result in a more stable calibration that is closer to the ideal one.

Then, the top two performing models will be used in the CJPE task to extract the embeddings and construct the explanations. To establish a baseline, the results obtained in [5] are considered and compared to the methods proposed in this thesis. The implementation code for this work is available on the GitHub repository at: `https://github.com/salvatorecurello/Master_Thesis.git`.

7.1 Experimental parameters

This section describes the training parameters employed in the experiments, along with the hardware utilized.

7.1.1 CJP

Non-hierarchical Transformers models

Table 7.1 shows the experimental parameters used. The value of the batch size in BigBird is decreased to fit the memory resources available. The optimizer used is AdamW [54] and a learning rate scheduling strategy is applied involving a linear warm-up phase followed by a linear decay. The number of steps for the warmup phase is equal to 1000 and the total number of training steps is given by the product of the total number of batches and the number of epochs. In this experimental phase, along with the embedding extraction, computational resources are provided by hpc@polito, which is a project of Academic Computing within the Department of Control and Computer Engineering at the Politecnico di Torino (http://www.hpc.polito.it). In particular, it has been used 1 nVidia Tesla V100 SXM2 with 32 GB - 5120 cuda cores.

Hyperparameter Transformers models			
Model	**Epochs**	**Learning Rate**	**Batch size**
BigBird	3	2e-5	4
RoBERTa	5	2e-5	6
LegalBERT	3	2e-5	6
CaseLawBERT	3	2e-5	6
LegalLSGBERT	3	2e-5	6
LED	3	2e-5	6
LegalLED	3	2e-5	6

Table 7.1: Table of the Hyperparameter used for the Transformers models in the CJP task.

Hierarchical Transformers models

In this case, all experiments were conducted using *Adam* [55] as the optimizer, with a learning rate set to 0.001, binary cross-entropy loss, and a total of 3 epochs. The only exceptions are in CaseLaWBERT, where the learning rate is adjusted to 0.01, and for RoBERTa, where the number of epochs is set to 5. Unlike the previous

phase, computational resources for this stage include the Tesla T4 GPU provided by Google Colab[1].

7.1.2 CJPE

All experiments concerning the computation of the attention weights are conducted using Adam as optimizer, with a learning rate set to 0.001, binary cross-entropy loss, and a total of 3 epochs. Instead, for the extraction of the explanability scores in the occlusions method, the hierarchical model trained in the previous CJP task is exploited. Furthermore, the computational resources for this stage include the Tesla T4 GPU provided by Google Colab.

7.2 Metrics

The goodness of each model is evaluated through different metrics that are used to compare the algorithms and make choices during the hyperparameter tuning phase.

7.2.1 CJP

Concerning the CJP task, the most common metrics for classification tasks are used. They can be described as follow:

- **Accuracy:** represents the number of correct predictions divided by the total number of predictions, i.e.:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.1}$$

- **Precision:** represents the number of true predicted positives over the total number of positives as follow:

$$P = \frac{TP}{TP + FP} \tag{7.2}$$

- **Recall:** represents the number of true predicted positives over the sum of true positives and false negatives, formally:

$$R = \frac{TP}{TP + FN} \tag{7.3}$$

[1]https://colab.google/

- **F1-score:** represents the harmonic mean between precision and recall and can be expressed as follow:

$$F1 = 2\frac{P * R}{P + R} \tag{7.4}$$

Here:

- TP indicates the *true positive*, i.e. instances that are actually positive and are correctly predicted as positive.
- TN indicates the *true negative*, i.e. instances that are actually negative and are correctly predicted as negative.
- FP indicates the *false positive*, i.e. instances that are actually negative and are incorrectly predicted as positive.
- FN indicates the *false negative*, i.e. instances that are actually positive and are incorrectly predicted as negative.

While accuracy is widely used for evaluation purposes, it may not always provide a realistic "picture" of the overall performance of a classification model, especially in cases with imbalanced datasets. Despite the dataset used in the experiments does not have a strong imbalance (as shown in Figure 4.2 in Section 4.1), the metric used for the evaluation of the models is a revised version of the F1-score, denoted as F1-score "macro". Essentially, the F1-score is computed for each class, and the F1 score macro is then the unweighted mean of them. Formally:

- for the positive class:

$$F1_{positive} = 2\frac{P_{positive} * R_{positive}}{P_{positive} + R_{positive}} \tag{7.5}$$

- for the negative class:

$$F1_{negative} = 2\frac{P_{negative} * R_{negative}}{P_{negative} + R_{negative}} \tag{7.6}$$

Then, the F1 score macro is equal to:

$$F1_{macro} = \frac{F1_{positive} + F1_{negative}}{2} \tag{7.7}$$

Reliability curve

A ML model built for a classification task provides as output estimated probabilities of the predictions, also called *confidences*. These confidences indicate the level of certainty that the model has regarding its predictions for the assigned labels. Calibration refers to the alignment between the predicted probabilities and the true likelihood of an event occurring. Essentially, if a model is well-calibrated not only makes accurate predictions but also provides confidences that align with the actual likelihood of those predictions [56]. One method to access the calibration of a model is the reliability curve [57]. The ideal calibration is represented by the diagonal, i.e. a 45-degree curve line from (0, 0) to (1, 1). The more the reliability curve closely follows the diagonal the higher the alignment between the predicted and actual probabilities. Points on the curve that are consistently below the ideal curve suggest that the model is under-confident[2]. This implies that the model is too conservative and may miss some true positives. In contrast, if the points are consistently above the ideal curve, the model is overconfident[3].

7.2.2 CJPE

This subsections presents the metrics used for the evaluation of the explanability model. Since there are no standard metrics for evaluating generated explanations, machine translation metrics are relied upon for evaluation. They aim to measure the overlap between the machine-made explanations and the legal experts' gold explanations.

Jaccard Similarity Given A, the set of unique terms in the first document, and B the set of unique terms in the second document, the Jaccard similarity is:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{7.8}$$

where $|A \cap B|$ represents the number of terms in common and $|A \cup B|$ is the total number of unique terms in both documents.

[2]When the model predicts a low probability of the positive class, the actual observed proportion of positive instances is higher than predicted.

[3]The model predicts a high probability of the positive class, the actual observed proportion of positive instances is lower than predicted.

Overlap-Min and Max Given A, the set of unique terms in the first document, and B the set of unique terms in the second document, the Overlap-Min is:

$$O_{min} = \frac{|A \cap B|}{min(|A|, |B|)} \tag{7.9}$$

while, the Overlap-Max is:

$$O_{max} = \frac{|A \cap B|}{max(|A|, |B|)} \tag{7.10}$$

ROUGE-N score It stands for *Recall-Oriented Understudy for Gisting Evaluation* [58]. It considers the number of matching text units between the generated text and the gold standard. It is typically used for summarization tasks and can be formalized as follows:

$$ROUGE - N = \frac{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)} \tag{7.11}$$

where n represents the n-gram's length, and $Counnt_{match}(gram_n)$ is the number of n-grams that appears in a candidate and a set of reference summaries. This formula can be applied to various values of N. During the experiments unigrams and bigrams are considered.

ROUGE-L score This metric measures the Longest Common Subsequence (LCS) between the reference and the candidate. In essence, given a candidate A and a reference B, the longest common sub-sequence of A and B is a common sub-sequence with maximum length. It is based on the intuition that the longer the match the higher the similarity among the summaries.

BLEU score It stands for Bilingual Evaluation Understudy [59] and it is typically used in machine translation. BLEU is calculated by dividing the number of matching words in the translated hypothesis sentence by the total word count in that hypothesis. It is a precision oriented metric where:

$$p_n = \frac{\sum_{C \in Candidates} \sum_{gram'_n \in C} Count_{match}(gram_n)}{\sum_{C' \in Candidates} \sum_{gram'_n \in C'} Count(gram'_n)} \tag{7.12}$$

Let c the candidate's length and r the length of reference corpus, a brevity penalty is computed as follows:

$$BP = \begin{cases} 1 & c > r \\ e^{(1-r/c)} & c \leq r \end{cases} \tag{7.13}$$

Then:

$$BLEU = BP * exp\left(\sum_{n=1}^{N} w_n log\, p_n\right) \tag{7.14}$$

where w_n are the weights assigned to different n-gram precisions.

METEOR It stands for *Metric for Evaluation of Translation with Explicit ORdering* [60]. METEOR takes into account not only exact word matches but also considers stemmed words, synonyms, and word order. It is based on the alignment between the generated and reference text.

Considering:

- The precision P as the number of unigrams in the candidate translation also found in reference over number of unigrams in the candidate translation.
- The recall R as the number of unigrams in the candidate translation also found in reference over the number of unigrams in the reference translation.

The METEOR score is defined as:

$$M = F_{mean}(1 - p) \tag{7.15}$$

where:

- p is the chunk penalty computed as $p = 0.5(\frac{Number\ of\ chunks\ in\ candidate}{Number\ of\ unigrams\ in\ candidate})$
- F_{mean} is the combination of precision and recall via a harmonic-mean computed as $\frac{10PR}{R+9P}$

7.3 Results

This section outlines an analysis derived from evaluating the results of the Court Judgment Prediction and Explanation task. It illustrates the quantitative results obtained according to the metrics described previously.

7.3.1 CJP

A parallel analysis is conducted, considering both $ILDC_{single}$ and $ILDC_{multi}$. All the base models are trained on the final section of the documents since it is the most relevant part that guides the prediction, as affirmed by the authors in [5]. In this section, are reported the top results based on F1-score. For exhaustive results, including all metrics and experiments, refer to Section A.1 of the Appendix.

Non-hierarchical Transformer models

In this phase, all the models are trained on the last N tokens of each document, where N is the maximum amount of tokens supported by the model.

In particular, in Table 7.2 are reported the results of each Transformer model trained on $ILDC_{single}$. As expected LegalLSGBERT reached the higher F1-score

Model	Tokens	Precision	Recall	F1-score
LegalBERT	512	72.80	68.39	70.53*
CaseLawBERT	512	71.92	68.13	69.97*
RoBERTa	512	72.21	61.46	64.04*
LED	1024	68.12	60.73	64.21*
LegalLED	1024	68.78	59.03	63.53*
LegalLSGBERT	3072	**78.33**	**76.91**	**77.61**
BigBird	4096	74.65	60.16	66.63*

Table 7.2: Metric scores obtained by training Transformers models on $ILDC_{single}$. The "Tokens" column refers to the number of "last" tokens of each document exploited. Performances marked with asterisks denote statistically significant differences indicating lower performance compared to the best-performing model in term of F1-score (LegalLSGBERT). This significant difference was calculated using a t-test with level of significance equal to 0.05.

among all the models thanks to its Local-Sparse-Global attention, its capacity to handle lengthy documents, and the information (legal terms) gained from its pre-training corpus. Consequently, a deeper exploration is performed to determine the optimal amount of "last" tokens. Figure 7.1 shows that the peak of performance occurs with 3072 tokens.

In contrast to what is stated by the authors in [5], all the domain-specific models (except LegalLED) outperform the generic ones, even though their pretraining corpora are composed of legal documents related a legal system that is completely different than the Indian one. This is further demonstrated by the higher performance of LegalBERT and CaseLawBERT compared to BigBird, despite using 6x fewer tokens.

Additonally, LegalLSGBERT outperformed the RoBERTa baseline model, which achieved a F1-score of 71.77%, as reported by the authors in [5]. This is noteworthy considering that their training was conducted on $ILDC_{multi}$, which comprises about 35,000 documents, compared to the 5,000 documents in $ILDC_{single}$.

Similar experiments are conducted considering the $ILDC_{multi}$. Also in this phase all the Transformer models are trained on the last N tokens of each document, where N represents the maximum number of tokens supported by the base model.

As reported in Table 7.3, LegalLSGBERT remains the top-performing model

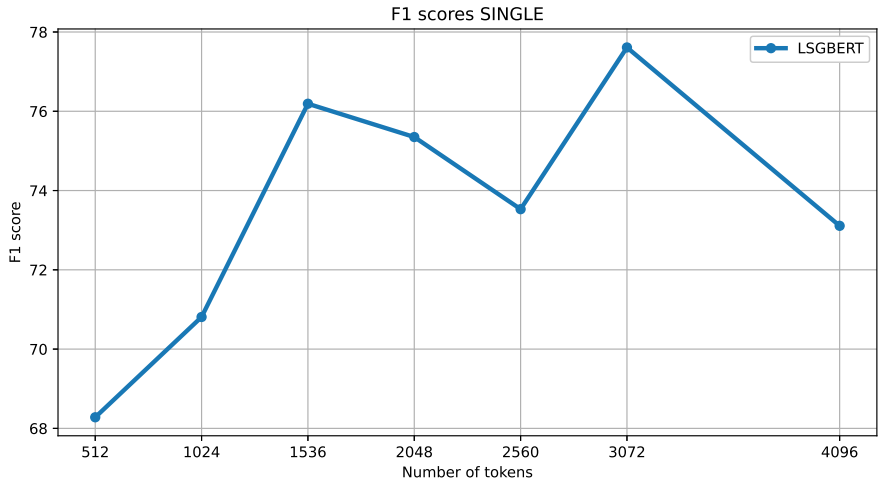


Figure 7.1: Exploration of the optimal amount of tokens for LegalLSGBERT trained on $ILDC_{single}$. The x-axis represents the number of "last" tokens considered, while the y-axis represents the corresponding F1 score obtained.

with an F1-score of 82.01%, outperforming the RoBERTa baseline model (71.77%) referring to the Transformers models and even the best configuration of the Hierarchical transformer Model (XLNet + BiGRU, 77.79%) obtained by the authors in [5]. In this case, its the optimal number of last tokens is determined to be 2560, as shown in Figure 7.2.

Model	Tokens	Precision	Recall	F1-score
CaseLawBERT	512	76.57	74.49	75.51*
LegalBERT	512	75.72	74.93	75.32*
RoBERTa	512	70.11	69.14	69.62*
LegalLED	1024	69.05	68.16	68.97*
LED	1024	68.45	67.36	67.90*
LegalLSGBERT	2560	**82.07**	**81.95**	**82.01**

Table 7.3: Metric scores obtained by training Transformers models on $ILDC_{multi}$. The "Tokens" column refers to the number of "last" tokens of each document exploited. Performances marked with asterisks denote statistically significant differences indicating lower performance compared to the best-performing model in terms of F1-score (LegalLSGBERT). This significant difference was calculated using a t-test with an alpha equal to 0.05.

In line with the previous observation, domain-specific models outperformed the generic ones, emphasizing what was stated in [45] and [44], i.e. generic models tend

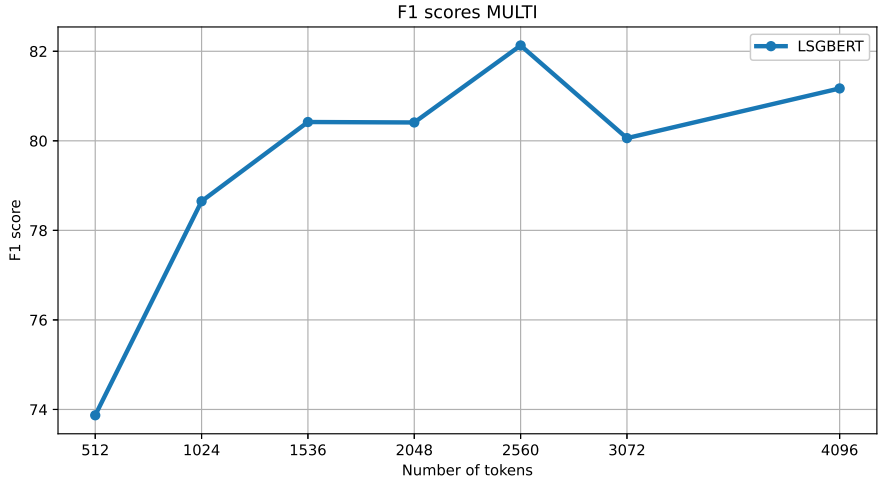


Figure 7.2: Exploration of the optimal amount of tokens for LegalLSGBERT trained on $ILDC_{multi}$. The x-axis represents the number of last tokens considered, while the y-axis represents the corresponding F1 score obtained.

to under-perform in specialized domains.

Furthermore, as expected, the results achieved through training on the $ILDC_{multi}$ are higher compared to those obtained from training on $ILDC_{single}$, due to the larger number of documents in the training set.

Hierarchical Transformer models

This subsection shows the results obtained by applying transformers hierarchically. When utilizing $ILDC_{single}$, only the top 3 performing base models are employed, except for LegalLSGBERT, which already considers a longer context.

The results reported in Table 7.4 shows that Hierarchical Transformer models bring a slight increase in performance.

Similar to the previous results, Domain-specific transformers (LegalBERT and CaseLawBERT) obtain higher F1-scores than the generic ones (RoBERTa). Furthermore, different Pooling strategies perform similarly, as shown in Figure 7.3.

By data augmenting the $ILDC_{single}$, the training set reaches ~38,000 samples, which is close to the number of samples obtained by using the last N token of each document in the $ILDC_{multi}$ (~35,000). As demonstrated later, using the $ILDC_{multi}$ in Hierarchical transformers yields better performance than augmenting the $ILDC_{single}$. This gives another intuition that the end of the documents is the part that primarily guides the predictions since, through data augmentation, the beginning and middle parts are included in the training set.

The application of Hierarchical Transformers using the last tokens of each

Model	Pooling strategy	Precision	Recall	F1-score
LegalBERT + BiGRU	[CLS]	77.51	76.21	76.86
LegalBERT + BiGRU	MEAN	76.99	76.83	76.91
LegalBERT + BiGRU	MAX	74.85	74.11	74.48
RoBERTa + BiGRU	[CLS]	75.14	74.64	74.89
RoBERTa + BiGRU	MEAN	75.22	73.89	74.56
RoBERTa + BiGRU	MAX	75.39	74.91	75.15
CaseLawBERT + BiGRU	[CLS]	**77.61**	**77.29**	**77.45**
CaseLawBERT + BiGRU	MEAN	75.83	75.29	75.56
CaseLawBERT + BiGRU	MAX	73.80	71.43	72.60

Table 7.4: Metric scores obtained using Hierarchical Transformer models. In this case, the $ILDC_{single}$ is augmented by dividing each document in chunks of 512 tokens with an overlap of 100 tokens. Then, to each chunk is assigned the same label of the entire document.

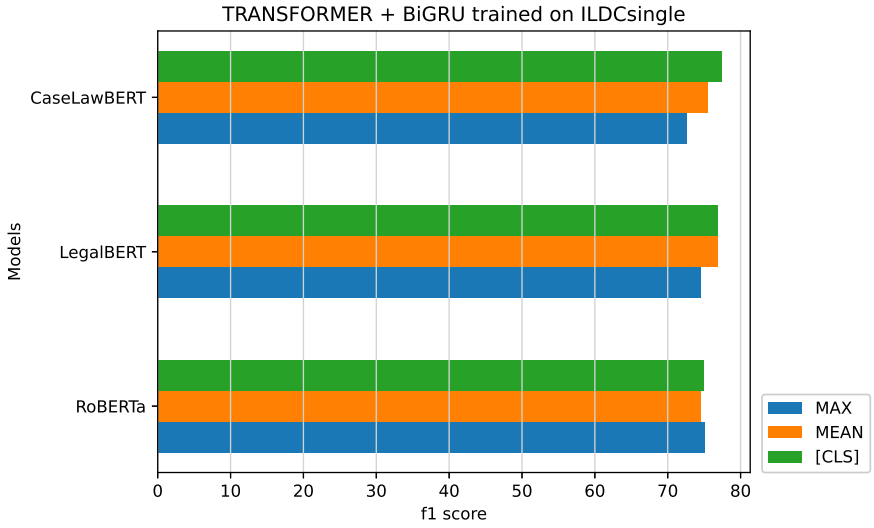


Figure 7.3: Hierarchical Transformer Models results obtained by data augmenting the $ILDC_{single}$.

document in $ILDC_{multi}$ further increased the performance. This is reported in Table 7.5, in which LegalLSGBERT emerges as top-performing model for the Court Judgement Prediction task obtaining an F1-score of 82.13, using the [CLS] token. It outperforms not only the previous results on Transformers models of Section 7.3.1, highlighting the superior efficacy of Hierarchical transformers, but also the XLNet + BiGRU baseline model which obtains a score of 77.79, according to [5].

Furthermore, each Hierarchical Transformer model outperforms the corresponding non-hierarchical version, and also in this scenario, Domain-specific transformers obtain higher performance than the generic ones.

Finally, Figure 7.4 illustrates that, among the three pooling strategies proposed, utilizing the [CLS] token proves to be the most effective strategy. However, the only exception is observed for LED and LegalLED. This drop can be justified by the fact that these models leverage a combination of global and local attention mechanisms (as described in Section 2.2.2). The [CLS] token is used as a representation of the entire sequence, but the intrinsic design of LED and LegalLED, emphasizing both global and local contextual information, implies a potential mismatch in prioritizing the importance of the [CLS] token. The global attention mechanism in LED and LegalLED ensures that the model captures contextual dependencies across the entire document. Meanwhile, the local attention, implemented through a sliding window, captures local relationships within smaller segments of the text. As a result, the [CLS] token, which is fundamentally a global representation, may not be optimally valued and even may lose its significance when applied in these models.

To summarize, Table 7.6 shows the best results obtained using both Non-hierarchical and Hierarchical Transformer models. The best configuration is achieved considering the $ILDC_{multi}$ with the hierarchical version of LegalLSG-BERT exploiting the [CLS] token.

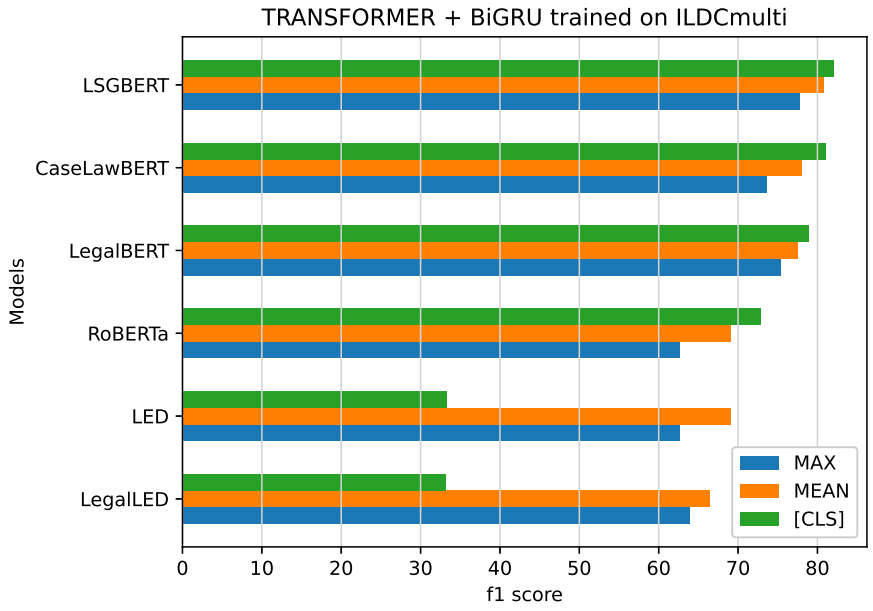


Figure 7.4: Hierarchical Transformer Models results obtained by using the last N tokens of each document in $ILDC_{multi}$.

Model	Pooling strategy	Precision	Recall	F1-score
LegalBERT	[CLS]	78.92	78.88	78.90
LegalBERT	MEAN	77.61	77.39	77.50
LegalBERT	MAX	75.43	75.31	75.37
LegalLED	[CLS]	32.34	33.33	33.25
LegalLED	MEAN	66.54	66.47	66.45
LegalLED	MAX	64.94	64.34	64.00
LED	[CLS]	32.33	33.32	33.29
LED	MEAN	61.31	67.72	69.05
LED	MAX	55.01	57.02	62.74
RoBERTa	[CLS]	72.78	72.33	72.55
RoBERTa	MEAN	70.42	67.73	69.06
RoBERTa	MAX	67.74	58.43	62.74
CaseLawBERT	[CLS]	81.15	81.07	81.12
CaseLawBERT	MEAN	78.01	77.97	78.00
CaseLawBERT	MAX	72.66	73.65	73.68
LegalLSGBERT	[CLS]	**82.07**	**82.06**	**82.13**
LegalLSGBERT	MEAN	80.86	80.81	80.84
LegalLSGBERT	MAX	77.84	77.83	77.84

Table 7.5: Metric scores obtained using Hierarchical Transformer models. In this case, the last tokens of each document in $ILDC_{multi}$ are used to fine-tune the base model. The results of the t-tests do not provide significant evidence that the best model is statistically better than other Hierarchical models. The configuration tested is LegalLSGBERT using the [CLS] token (best model) against all others hierarchical Transformers exploiting the [CLS] token, since it turned out to be the best pooling strategy

Model	Corpus	Precision	Recall	F1-score
LegalLSGBERT	single	78.33	76.91	77.61
LegalLSGBERT	multi	82.07	81.95	82.01
CaseLawBERT+BiGRU	single	77.61	77.29	77.45
LegalLSGBERT+BiGRU	multi	**82.07**	**82.06**	**82.13**

Table 7.6: Best scores obtained in both Non-hierarchical and Hierarchical Transformer models.

7.3.2 Statistical Analysis of Results

The goal of this analysis is to discern, through the use of statistical methods, whether observed variations in outcomes can be attributed to genuine model differences

rather than random chance.

Firstly, the analysis focus on analysing whether there are statistical differences between Domain-specific Transformers and Generic Transformers, validating the insights obtained during the experiments. The type of test used is a "one-sided" *Dependent t-test for paired samples* that identifies if the mean of the predictions' distribution generated by a model (expected to outperform) is significantly greater than the mean of the predictions' distribution produced by another model.

The results presented in Table 7.7 confirm the initial assumption. The tests reveal statistically significant differences in the performance of the two models, confirming that the Domain-specific ones are indeed superior.

"Domain" vs "Generic" t-Test Analysis			
Generic	**Domain-specific**	**p-value**	**Statistical Significance**
BERT	LegalLSGBERT	< 0.05	There are statistically significant differences
	CaseLawBERT	< 0.05	There are statistically significant differences
	LegalBERT	< 0.05	There are statistically significant differences
RoBERTa	LegalLSGBERT	< 0.05	There are statistically significant differences
	CaseLawBERT	> 0.05	There are not sufficient evidence to affirm that there are statistically significant differences
	LegalBERT	< 0.05	There are statistically significant differences

Table 7.7: Comparison of Domain-Specific Transformers and Generic Transformers using t-Test Analysis with level of significance equal to 0.05. Notably, the predictions of the BERT model are computed using the pretrained model provided by the author in [5].

These tests are also conducted to determining the presence of statistical differences between Hierarchical Transformers and their corresponding non-hierarchical versions. The goal is to assess the effectiveness of applying transformers hierarchically.

Table 7.8 shows the results obtained. Essentially, the t-test demonstrates that all the p-values are lower than α, implying that all hierarchical models are indeed superior to the standard transformer models.

7.3.3 Calibration study

This section describes a calibration study conducted on the top two performing models:

- LegalLSGBERT+BiGRU which achieved an F1-score of 82.13, using the [CLS] token.
- CaseLawBERT+BiGRU, instead, achieved an F1-score of 81.12 also utilizing using the [CLS] token.

"Hier. Transformers" t-Test Analysis		
Model	**p-value**	**Statistical Significance**
LegalLSGBERT	< 0.05	There are statistically significant differences
CaseLawBERT	< 0.05	There are statistically significant differences
LegalBERT	< 0.05	There are statistically significant differences
RoBERTa	< 0.05	There are statistically significant differences

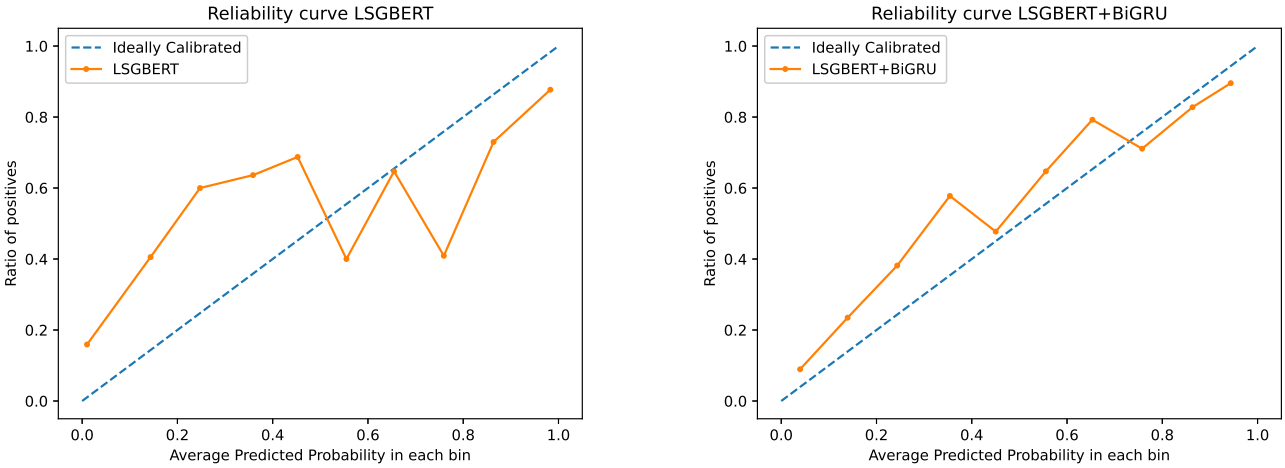
Table 7.8: Results of the t-test of each Transformer models compared to the corresponding hierarchical version. The level of significance is set to 0.05.

Figure 7.5 shows the reliability curves of LegalLSGBERT and LegalLSG-BERT+BiGRU. A visual analysis of the figures demonstrated that the hierarchical transformer offers a better calibration, probably due to its internal design that better captures dependencies at multiple levels and the information flow among the layers that helps the model to refine its predictions at different levels of granularity. Similar behavior is shown by CaseLawBERT, as illustrated in Figure 7.6.

To further investigate this observation the histograms of the predictions are visualized in Figures 7.7 and 7.8. In these plots, the x-axis is divided into intervals or bins, each corresponding to a specific range of the predicted probabilities while the y-axis represents the frequency or count of predictions falling into each bin. These plots can provide insights into whether the model is assigning more balanced probabilities. In particular, in the hierarchical model (Figure 7.8b) a distinct V-shape is evident. The model tends to assign lower probabilities when uncertain and higher probabilities when more confident. This can be a sign of a good calibration since the predicted probabilities align well with the actual probability of belonging to a class. The V-shaped histogram suggests that the model is less likely to exhibit systematic overestimation or underestimation, and probabilities are distributed more evenly. On the other hand, in the plot associated with the standard Transformer model, the V-shape is narrower (Figure 7.8a). This could indicate that the model is overconfident in its predictions, potentially leading to misleading results. However, similar behavior is obtained with LegalLSGBERT (Figure 7.7a and Figure 7.7b). While the difference is less pronounced, it aligns with the behavior observed in the reliability curves, where CaseLawBERT exhibits superior calibration.

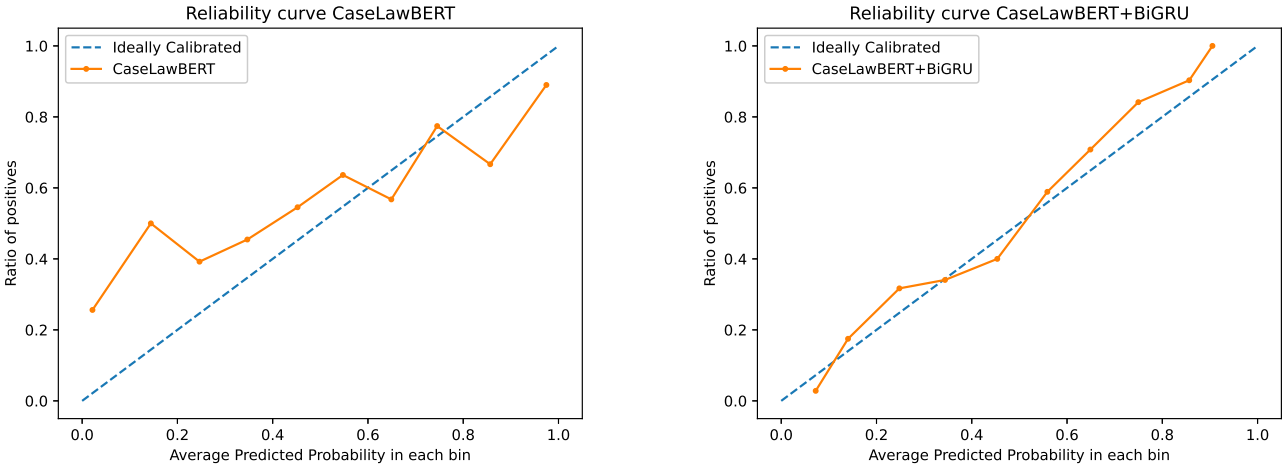
7.3.4 CJPE

This section focuses on illustrating the results achieved by measuring the overlap between the explanation generated through the explainability model and the manually annotated explanations contained in $ILDC_{expert}$. Similar to the Court



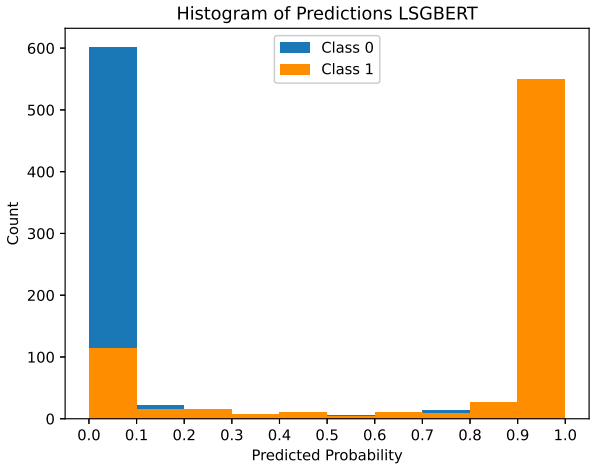
(a) LegalLSGBERT **(b)** LegalLSGBERT+BiGRU

Figure 7.5: Reliability curves of LegalLSGBERT. Figure (a) refers to the non-hierarchical version of LegalLSGBERT (82.01% F1-score), while Figure (b) refers to the hierarchical LegalLSGBERT (82.13% F1-score).

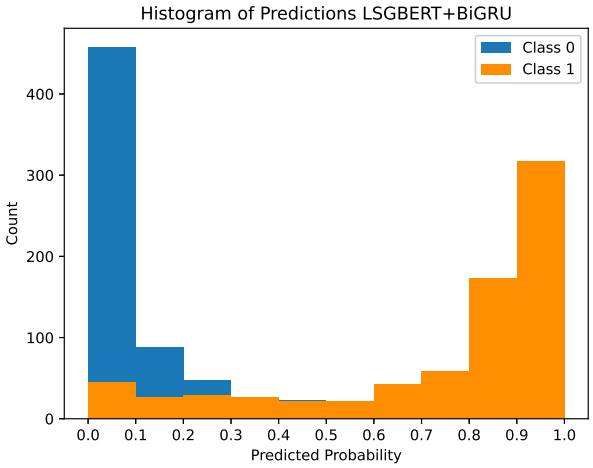


(a) CaseLawBERT **(b)** CaseLawBERT+BiGRU

Figure 7.6: Reliability curves of CaseLawBERT. Figure (a) refers to the non-hierarchical version of CaseLawBERT (75.51% F1-score), while Figure (b) refers to the hierarchical CaseLawBERT (81.12% F1-score).

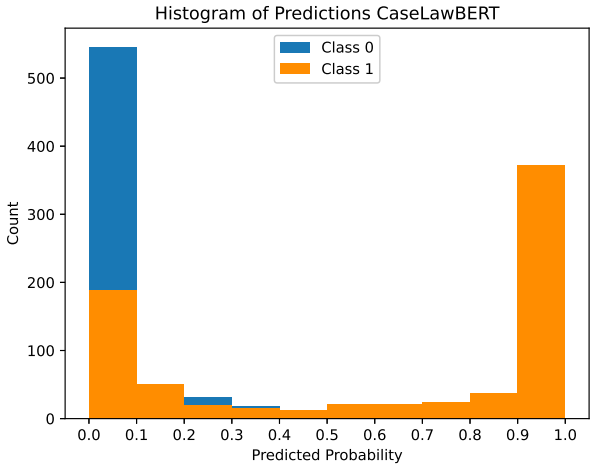


(a) LegalLSGBERT

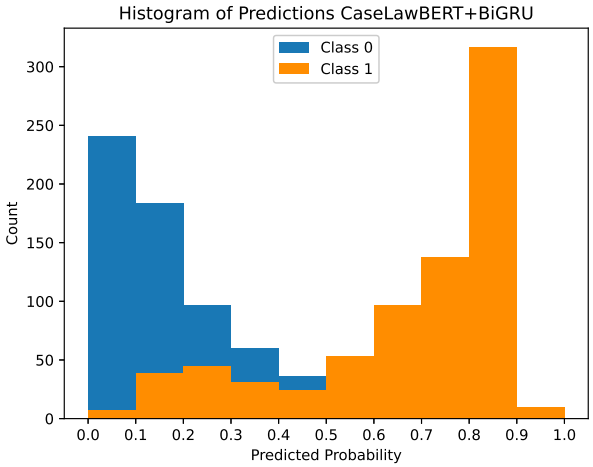


(b) LegalLSGBERT+BiGRU

Figure 7.7: Histograms of the predictions of LegalLSGBERT. Figure (a) refers to the non-hierarchical version of LegalLSGBERT (75.51% F1-score), while Figure (b) refers to the hierarchical LegalLSGBERT (82.13% F1-score).



(a) CaseLawBERT



(b) CaseLawBERT+BiGRU

Figure 7.8: Histograms of the predictions of CaseLawBERT. Figure (a) refers to the non-hierarchical version (82.01% F1-score), while Figure (b) refers to the hierarchical one (81.12% F1-score).

Judgment Prediction task, the baseline considered is the results obtained by the authors in [5]. Notably, the experiments are conducted using the top two performing models: LegalLSGBERT+BiGRU and CaseLawBERT+BiGRU (both exploiting the [CLS] token). The results are obtained by using both occlusions method and attention mechanism. In particular, the attention mechanism has been examined considering the original chunk division of the corpus (as in [5]) and a revised division that ensures the last chunk is complete.

Occlusions method

The quantitative results using the occlusions method are computed by considering both LegalLSGBERT and CaseLawBERT. The scores obtained are reported in Table 7.9 and Table 7.10, respectively. The explanations are generated by considering the top k% sentences with $k = \{20,30,40,50\}$. For space constraints and a fair comparison with the results achieved in [5], the tables report the scores considering the top 40% most important sentences. Refers to Appendix A.2 for the quantitative results achieved using different values of k.

Metric (40% of sentences)	**Explainability Model vs Expert**					
	Expert					
	1	**2**	**3**	**4**	**5**	**Mean**
Jaccard Similarity	0.334	0.313	0.330	0.329	0.319	0.325
Overlap-Min	0.687	0.573	0.739	0.770	0.596	0.673
Overlap-Max	0.406	0.415	0.378	0.372	0.417	0.398
ROUGE-1	0.494	0.474	0.483	0.495	0.482	0.486
ROUGE-2	0.335	0.292	0.338	0.350	0.313	0.326
ROUGE-L	0.476	0.442	0.469	0.486	0.456	0.466
BLEU	0.176	0.282	0.115	0.111	0.273	0.192
Meteor	0.215	0.302	0.177	0.174	0.285	0.231

Table 7.9: Explanation generation results using the Occlusions method with LegalLSGBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 40.

Both tables have similar results, meaning that there are not many differences in using LegalLSGBERT or CaseLawBERT. Notably, for the embedding extraction the model related to LegalLSGBERT was trained on 2560 tokens while for CaseLawBERT only 512 tokens were required.

As in [5], the highest overlap values is obtained with *Expert*4. In particular, the Overlap-Min demonstrates high agreements between the model and the legal experts. However, the other metrics are in the low to medium range, ROUGE-1 is

Metric (40% of sentences)	Explainability Model vs Expert					
	Expert					
	1	2	3	4	5	Mean
Jaccard Similarity	0.345	0.323	0.328	0.329	0.307	0.325
Overlap-Min	0.685	0.576	0.727	0.759	0.559	0.663
Overlap-Max	0.423	0.430	0.382	0.375	0.409	0.404
ROUGE-1	0.513	0.488	0.484	0.497	0.470	0.490
ROUGE-2	0.358	0.320	0.342	0.356	0.302	0.336
ROUGE-L	0.500	0.465	0.471	0.489	0.446	0.474
BLEU	0.200	0.322	0.136	0.130	0.284	0.214
Meteor	0.228	0.325	0.189	0.187	0.291	0.244

Table 7.10: Explanation generation results using the Occlusions method with CaseLawBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 40.

the highest achieving on average a score of 0.490. This can be justified by the fact that metrics such as ROUGE-2, Jaccard Similarity, BLUE (only partially), and METEOR are influenced by the order of the words. This aspect is crucial since, as stated in [5], this method is not capable of performing a rank of the extracted sentences. In contrast, the golden explanations of the legal experts are ranked from 1 to 10.

The majority of the metrics slightly outperform the baseline results obtained in [5], demonstrating that the embedding extractions using Domain-specific transformers lead to better performances. This can be seen in Table 7.11 highlighting how, even when using the same methodologies, models that are more explainable are obtained.

Finally, it is observed that reducing the value of k results in a decrease across all metrics, except for Overlap-Min, which records higher scores as the length of the explanation decreases. This phenomenon may be attributed to the term in the denominator becoming smaller, leading to division by a reduced value.

Attention Mechanism

Concerning the attention mechanism, the results are reported in Table 7.12 and Table 7.13, for LegalLSGBERT and CaseLawBERT respectively. In this scenario, the results indicate a drop in performance, primarily caused by the relatively smaller number of tokens in the generated explanation, averaging 185 compared to the 730 tokens generated for the occlusions methods. The shorter length of the explanation is attributed to the reduced size of the last chunk in each document,

Model	Jaccard Similarity	Overlap Min	Overlap Max	R1	R2	RL	BLUE	Meteor
Baseline	0.324	**0.719**	0.383	0.451	0.297	0.424	0.176	0.231
LegalLSGBERT	**0.325**	0.673	0.398	0.486	0.326	0.466	0.192	0.231
CaseLawBERT	**0.325**	0.663	**0.404**	**0.490**	**0.336**	**0.474**	**0.214**	**0.244**

Table 7.11: Comparison between the baseline and the explainability models obtained using LegalLSGBERT and CaseLawBERT. The results represent the average scores among the five legal experts obtained using the occlusions method. As baseline the results in [5] are considered.

as it is often not complete in most cases. In particular, by extracting the attention weight of each chunk it is possible to understand which is the most important chunk of the document. A visual analysis can be conducted by plotting the mean attention chunk scores. This involves averaging the chunk scores of the documents, contained in the test set(1517 documents), having a same number of chunks. For space constraints, it is not possible to display all the plots, but an example is reported in Figure 7.9. This figure shows average attention scores among the documents of the test set that are represented by 13 chunks. It is clearly visible that higher scores are assigned to the last chunks, confirming the intuition that the most relevant syntactic and semantic information lies in the last section of the document. However, this behavior is reported also in other documents represented by a different number of chunks.

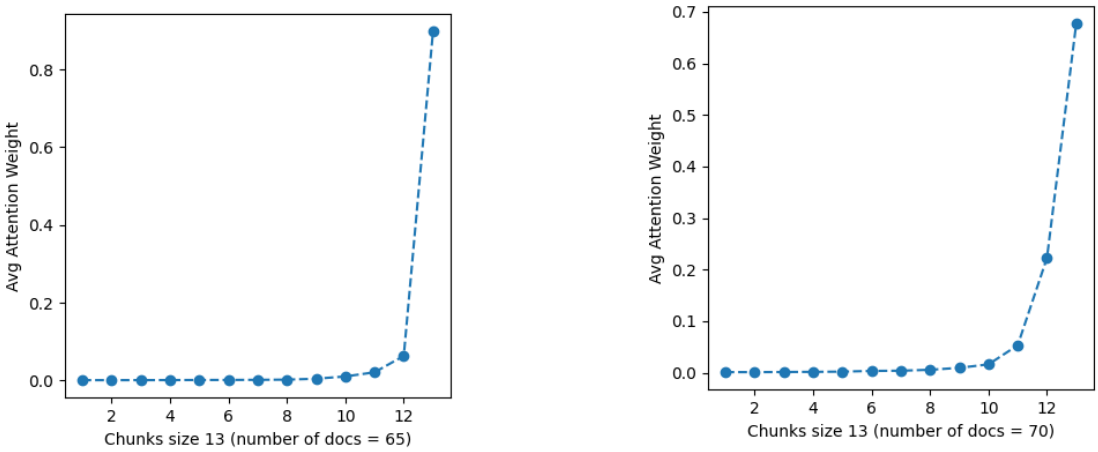
To confirm this intuition, a similar analysis is performed using the occlusion scores. As shown in Figure 7.10, the behavior is similar to the attention mechanism. Higher occlusion scores are assigned to the chunks representing the end of the documents. This is a further confirmation that the end of the document is the most relevant part of the document.

Despite all the documents of the test set being characterized by having the last chunk as the most representative one (i.e. it has a higher attention score), some documents do not show this behavior. A deeper analysis of the attention weights demonstrated these specific documents have in common these characteristics:

- Their most representative chunk is the penultimate one.
- Typically, the last chunk contains many fewer tokens compared to the previous one.

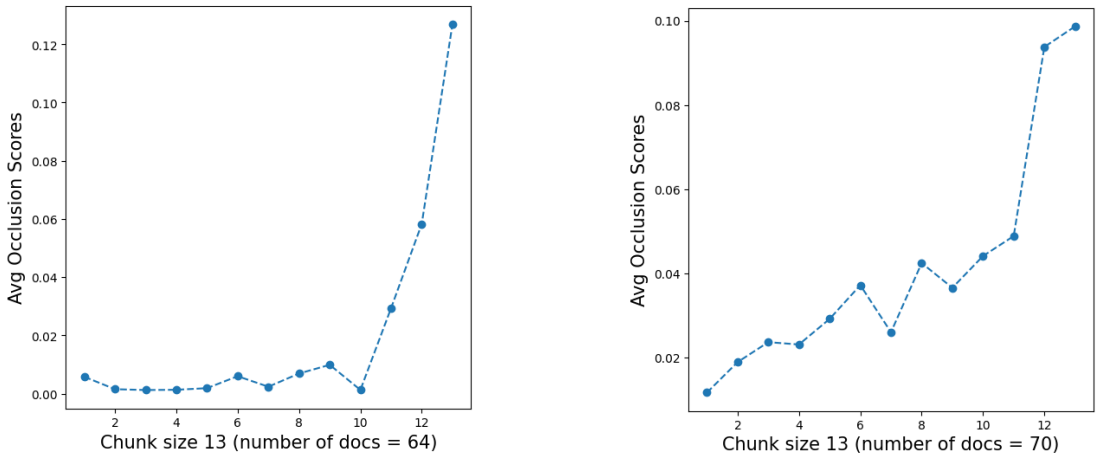
This scenario has been reported in ~3% of the documents in the test set.

To address this issue, it is recommended to consider a different chunk division for the corpus. Initiating the splitting process from the end ensures that all the



(a) LegalLSGBERT **(b)** CaseLawBERT

Figure 7.9: The average attention scores relate specifically to the documents of the test that are are represented by 13 chunks. Figure (a) refers to the chunks extracted with LegalLSGBERT, while Figure (b) refers to the chunks extracted with CaseLawBERT.



(a) LegalLSGBERT **(b)** CaseLawBERT

Figure 7.10: The average occlusion scores related specifically to the documents of the test set that are are represented by 13 chunks. Figure (a) refers to the chunks extracted with LegalLSGBERT, while Figure (b) refers to the chunks extracted with CaseLawBERT.

last chunks of each document are fully occupied. The results obtained by using these methods are reported in Table 7.14 for LegalLSGBERT and Table 7.15 for CaseLawBERT.

Metric	Explainability Model vs Expert					
	Expert					
	1	2	3	4	5	Mean
Jaccard Similarity	0.193	0.220	0.161	0.160	0.201	0.187
Overlap-Min	0.806	0.678	0.808	0.840	0.691	0.765
Overlap-Max	0.206	0.256	0.170	0.168	0.230	0.206
ROUGE-1	0.287	0.324	0.245	0.251	0.300	0.281
ROUGE-2	0.181	0.120	0.149	0.156	0.177	0.173
ROUGE-L	0.279	0.305	0.236	0.250	0.283	0.269
BLEU	0.035	0.100	0.020	0.018	0.085	0.052
METEOR	0.091	0.152	0.073	0.072	0.143	0.106

Table 7.12: Explanation generation results using the attention mechanism with LegalLSGBERT. The "Mean" column represents the average scores computed among the five legal experts. The chunk division of the corpus is performed starting from the beginning.

A comparison of the results are reported in Table 7.16 in which all the metrics, except Overlap-Min, show higher performance with respect to the original approach.

Finally, Figure 7.11 shows a comparison of all the methods used, considering CaseLawBERT (similar behavior is achieved by LegalLSGBERT). Essentially, ensuring that the last chunk extracted from the documents is fully complete leads to improved performance if using the attention mechanism, as the crucial part of the explanation is not split between the last chunk and the penultimate one.

The average number of tokens in the explanations increases compared to the previous chunk division approach. While the performance is slightly lower than the occlusions method, the advantage lies in shorter explanations. In some cases, this can facilitate the rapid delivery of insights to expedite the decision-making process.

7.4 Revised $ILDC_{expert}$

An essential aspect when evaluating explanations generated by a model and those generated by a human is that the source document is identical for both. An in-depth analysis of the ILDC corpus has revealed inconsistencies between the input text received by the model and that provided to the legal expert. This analysis revealed that during the preprocessing applied in the dataset's data scraping process, some

Metric	Explainability Model vs Expert					
	Expert					
	1	2	3	4	5	Mean
Jaccard Similarity	0.238	0.253	0.197	0.195	0.245	0.226
Overlap-Min	0.811	0.676	0.813	0.842	0.714	0.771
Overlap-Max	0.254	0.298	0.210	0.205	0.281	0.250
ROUGE-1	0.345	0.368	0.296	0.301	0.358	0.334
ROUGE-2	0.232	0.235	0.192	0.197	0.231	0.217
ROUGE-L	0.337	0.35	0.287	0.295	0.342	0.322
BLEU	0.055	0.141	0.036	0.032	0.124	0.078
METEOR	0.116	0.186	0.096	0.092	0.180	0.134

Table 7.13: Explanation generation results using the attention mechanism with CaseLawBERT. The "Mean" columns represents the average scores computed among the five legal experts. The chunk division of the corpus is performed starting from the beginning.

Metric	Explainability Model vs Expert					
	Expert					
	1	2	3	4	5	Mean
Jaccard Similarity	0.345	0.330	0.288	0.292	0.343	0.320
Overlap-Min	0.792	0.629	0.788	0.831	0.695	0.747
Overlap-Max	0.381	0.415	0.314	0.312	0.408	0.366
ROUGE-1	0.474	0.467	0.411	0.427	0.478	0.451
ROUGE-2	0.344	0.315	0.288	0.306	0.335	0.318
ROUGE-L	0.463	0.443	0.398	0.419	0.459	0.436
BLEU	0.140	0.274	0.084	0.080	0.240	0.164
METEOR	0.196	0.288	0.156	0.154	0.279	0.215

Table 7.14: Explanation generation results using the attention mechanism with LegalLSGBERT. The "Mean" columns represents the average scores computed among the five legal experts. The chunk division of the corpus is performed starting from the end.

regular expressions are applied to convert some words to their abbreviated form. The following are some examples:

- "no." is converted to "number"
- "nos." is converted to "numbers"
- "co." is converted to "company"

Metric	Explainability Model vs Expert					
	Expert					
	1	**2**	**3**	**4**	**5**	**Mean**
Jaccard Similarity	0.349	0.33	0.293	0.297	0.345	0.323
Overlap-Min	0.787	0.625	0.784	0.827	0.693	0.743
Overlap-Max	0.387	0.416	0.32	0.318	0.41	0.370
ROUGE-1	0.482	0.47	0.419	0.435	0.483	0.458
ROUGE-2	0.352	0.317	0.295	0.314	0.339	0.323
ROUGE-L	0.471	0.449	0.406	0.428	0.464	0.444
BLEU	0.15	0.276	0.091	0.087	0.244	0.170
METEOR	0.203	0.292	0.161	0.159	0.282	0.219

Table 7.15: Explanation generation results using the attention mechanism with CaseLawBERT. The "Mean" columns represents the average scores computed among the five legal experts. The chunk division of the corpus is performed starting from the end.

Model	Jaccard Similarity	Overlap Min	Overlap Max	R1	R2	RL	BLUE	Meteor
Baseline	**0.324**	0.719	0.383	0.451	0.297	0.424	**0.176**	**0.231**
LegalLSGBERT	0.187	0.765	0.206	0.281	0.173	0.269	0.052	0.106
LegalLSGBERT(end)	0.320	0.747	0.366	0.451	0.318	0.436	0.164	0.215
CaseLawBERT	0.226	**0.771**	0.250	0.334	0.217	0.322	0.078	0.134
CaseLawBERT(end)	0.323	0.743	**0.370**	**0.458**	**0.323**	**0.444**	0.170	0.219

Table 7.16: Comparing results achieved by implementing a chunk division starting from both the beginning and the end. These results reflect the average scores from the occlusions method among the five legal experts. The notation '(end)' indicates that the chunk division starts from the end of the document.

- "ltd." is converted to "limited"

The main issue is that the annotators were provided with a text without these conversions. Considering that these are highly frequent words in the documents, and given that metrics operate at the token level, the overlap calculated by the metrics treats these words as distinct, even though they refer to the same entity. More critically, the text provided to annotators contained additional sentences compared to the input received by the model. These extra sentences, usually positioned at the end of the document, are highly relevant to the explanation and were almost consistently incorporated into the explanations manually annotated by the annotators. This has resulted in an evident decline in performance. For these

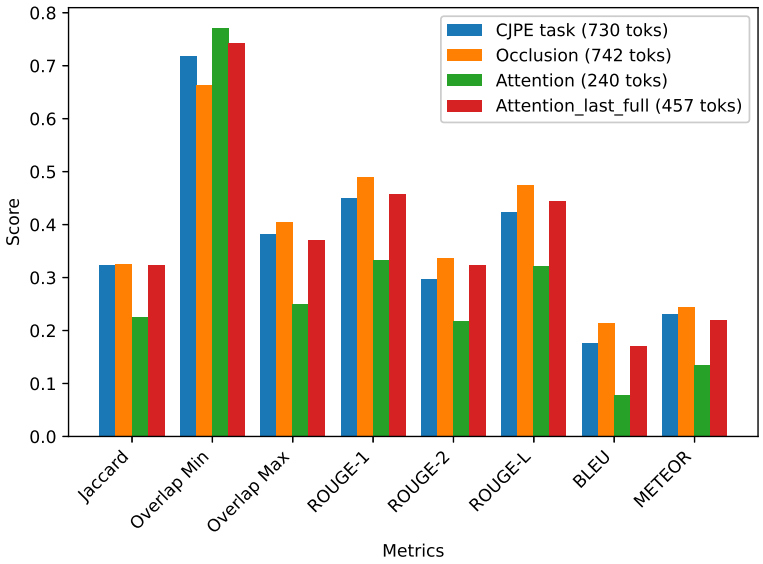


Figure 7.11: Barplot that illustrates the performance of the methods proposed. The embedding extraction is performed using CaseLawBERT exploiting the [CLS] token. The "CJPE task" refers to the results achieved in [25].

reasons, a new optimized version of the $ILDC_{expert}$ is created and all the metrics are re-computed. The results are shown in Table 7.17 for the occlusions method, in Table 7.18 for the attention mechanism with the original chunk division, and in Table 7.19 for the chunk division that starts from the beginning of the document. As expected, this small preprocessing of the corpus improved the performance of all methods, highlighting the critical importance of ensuring that the input data provided to both the model and annotators are as congruent as possible.

Metric (40% of the sentences)	Explainability Model vs Expert				
	Expert				
	1	**2**	**3**	**4**	**5**
Jaccard Similarity	0.362	0.363	0.369	0.366	0.361
Overlap-Min	0.734	0.619	0.804	0.831	0.642
Overlap-Max	0.426	0.473	0.411	0.400	0.459
ROUGE-1	0.532	0.534	0.538	0.548	0.534
ROUGE-2	0.392	0.372	0.412	0.420	0.380
ROUGE-L	0.517	0.505	0.528	0.541	0.512
BLEU	0.212	0.374	0.148	0.136	0.327
METEOR	0.243	0.353	0.204	0.195	0.326

Table 7.17: Explanation generation results using the Occlusions method with LegalLSGBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 40.

Metric	Explainability Model vs Expert					
	Expert					
	1	**2**	**3**	**4**	**5**	Mean
Jaccard Similarity	0.220	0.242	0.174	0.173	0.238	0.209
Overlap-Min	0.915	0.763	0.899	0.948	0.808	0.867
Overlap-Max	0.227	0.273	0.178	0.175	0.262	0.223
ROUGE-1	0.324	0.352	0.267	0.274	0.343	0.312
ROUGE-2	0.228	0.229	0.177	0.186	0.232	0.210
ROUGE-L	0.319	0.336	0.262	0.272	0.331	0.304
BLEU	0.036	0.099	0.013	0.012	0.099	0.052
METEOR	0.099	0.154	0.069	0.069	0.153	0.109

Table 7.18: Explanation generation results using the attention mechanism with LegalLSGBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The chunk division of the corpus is performed starting from the beginning.

Metric	Explainability Model vs Expert					
	Expert					
	1	**2**	**3**	**4**	**5**	**Mean**
Jaccard Similarity	0.408	0.388	0.335	0.342	0.416	0.377
Overlap-Min	0.874	0.698	0.867	0.917	0.785	0.828
Overlap-Max	0.432	0.468	0.353	0.353	0.467	0.415
ROUGE-1	0.548	0.53	0.472	0.494	0.555	0.520
ROUGE-2	0.444	0.397	0.368	0.394	0.437	0.408
ROUGE-L	0.539	0.507	0.462	0.49	0.537	0.507
BLEU	0.186	0.335	0.112	0.109	0.307	0.210
METEOR	0.233	0.336	0.181	0.182	0.332	0.253

Table 7.19: Explanation generation results using the attention mechanism with LegalLSGBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The chunk division of the corpus is performed starting from the end.

Chapter 8

Conclusions

The primary objectives of this thesis centered on two closely connected tasks: Court Judgment Prediction and Court Judgment Prediction and Explanation. The first involves predicting legal outcomes by capturing patterns in historical cases, while the second aims to provide transparent and comprehensible explanations for the predicted decisions. The main contribution of this work is the implementation of an automated system capable of performing these sub-tasks.

The extensive experimentation of Transformer models in the Court Judgment Prediction task has demonstrated the efficacy of Domain-specific transformers even when applied to legal systems different from the ones on which they were originally trained. Additionally, the experiments indicated that training the models on the last section of the document yielded higher performances. This implies that the end of the legal documents holds a substantial influence on the prediction outcomes. Furthermore, this thesis highlighted that hierarchical models, particularly when employing the [CLS] token for pooling at the document level, further improved the performances. This improvement suggests that capturing the overall context, through the [CLS] token, played a crucial role in the predictive process. They also provided a better prediction calibration that further enhanced the reliability of the model's predictions.

Regarding the Court Judgment Prediction and Explanation task, the occlusions method and attention mechanisms were instrumental in extracting relevant sentences from case descriptions. In particular, the occlusion method provides longer explanations, attempting to rank the most important sentences in the document. On the other hand, the attention mechanism exhibits similar performance to the occlusion method but delivers shorter explanations. In some cases, this brevity can be an advantage, offering a quick access to the needed information.

8.1 Future Directions

The results obtained provide motivation for further investigation in this field. The best prediction model implemented in this thesis has an accuracy of 82% versus 94% [5] for human legal experts. Potential improvements could involve utilizing transformers pretrained specifically with an Indian legal document corpus. While the transformer models exploited in the experiments were already pretrained with US/EU legal documents, the use of models pretrained with Indian legal documents may help capture patterns influenced by different cultural nuances or biases, potentially leading to better accuracy of the model.

Furthermore, taking into account temporal dynamics can be crucial. This aspect, denoted as *"temporal concept drift"* [61], can be significant in the legal domain since legal systems often evolve, and the context of legal decisions may change over time.

When it comes to explainability models, the situation is particularly critical. Despite the methods observed in this thesis partially improving the alignment between machine-generated and manually annotated explanations, there is still a wide gap to be bridged. The understanding and interpretation of the models used to explain predictions in legal contexts are subject to a significant disparity that requires considerable effort to overcome. A possible improvement can be to extend the hierarchical attention model for datasets that contains multi-modal information such as images, audio, or additional contextual information. This could be relevant in legal cases where the case description is accompanied by additional materials that may contribute to the explanation of the predicted decision. However, a significant challenge in this context lies in defining metrics that accurately measure the quality of the generated explanations.

Appendix A

Additional Results

A.1 CJP task

This section presents the results of all the experiments regarding the CJP task. In particular, Table A.1 refers to the application of Non-hierarchical Transformer models trained on the last tokens of each document in $ILDC_{single}$. The scores describe how using a model pretrained on the legal domain leads to better performance. The utilization of Transformers capable of handling long input sequences further improves the results. This is demonstrated by LegalLSGBERT, which is identified as the best-performing model. Similar considerations apply when training is performed on $ILDC_{multi}$. As shown in Table A.2, the results are even higher, given that the training set contains six times more documents. Table A.3, on the other hand, pertains to the Hierarchical Transformers model. In the majority of cases, exploiting the [CLS] token proves to be the most effective strategy. The optimal number of epochs for the aggregation model is 3 for all models except CaseLawBERT, which achieves its best performance with only 2 epochs. The highest-performing model in this task is LegalLSGBERT+BiGRU, achieving an F1-score of 82%, compared to the 94% accuracy achieved by legal experts.

A.2 CJPE task

This section presents the complete results obtained in all CJPE tasks. Specifically, Tables A.4, A.5, and A.6 display the outcomes achieved using LegalLSGBERT in combination with the occlusions method across various values of k. The scores indicate that reducing the value of k leads to lower performance but results in shorter explanations. The same scenario is found when using the revised version of the $ILDC_{expert}$, as shown in Tables A.10, A.11, and . Furthermore, Tables A.7, A.8, and A.9 refer to CaseLawBERT. Here, the performances slightly increased

compared to LegalLSGBERT, but, similarly, the results are higher for higher values of k. Finally, Tables A.13, A.14, and A.15 report the results concerning the revision of $ILDC_{expert}$ using CaseLawBERT. As expected, using this new version of the corpus leads to higher agreement with legal experts.

Model	**Tokens**	**Batch**	**E**	**A**	**P**	**R**	**F1**
LegalLSGBERT	512	6	3	66.71	72.21	66.83	69.41
LegalLSGBERT	1024	6	3	68.02	73.70	68.14	70.81
LegalLSGBERT	1536	6	3	75.34	76.70	75.34	76.19
LegalLSGBERT	2048	6	3	73.17	77.56	73.26	75.35
LegalLSGBERT	2560	4	3	71.12	75.98	71.22	73.53
LegalLSGBERT	3072	2	3	76.86	78.33	76.91	77.61
LegalLSGBERT	4096	2	3	70.00	76.38	70.11	73.11
LegalBERT	512	6	3	68.29	72.80	68.39	70.53
CaseLawBERT	512	6	3	68.03	71.92	68.13	69.97
BigBird	4096	2	3	59.99	74.65	60.16	66.63
LED	1024	6	3	60.58	68.12	60.73	64.21
RoBERTa	512	6	3	61.30	72.21	61.46	64.04
LegalLED	1024	6	3	58.87	68.78	59.03	63.53

Table A.1: Complete results obtained by training Transformers models on $ILDC_{single}$. The "Tokens" column refers to the number of "last" tokens of each document exploited. The "E" column refers to the number of epochs. The columns "A", "P", and "R" refers to accuracy, precision, and recall, respectively.

Model	**Tokens**	**Batch**	**E**	**A**	**P**	**R**	**F1-score**
LegalLSGBERT	512	6	3	73.30	74.40	73.35	73.87
LegalLSGBERT	1024	6	3	78.31	78.94	78.35	78.65
LegalLSGBERT	1536	6	3	80.22	80.59	80.24	80.42
LegalLSGBERT	2048	6	3	80.09	80.71	80.12	80.41
LegalLSGBERT	2560	6	3	81.93	82.07	81.95	82.01
LegalLSGBERT	3072	4	3	79.43	80.61	79.48	80.06
LegalLSGBERT	4096	2	3	80.61	81.68	80.66	81.17
CaseLawBERT	512	6	3	74.42	76.57	74.49	75.51
CaseLawBERT	512	6	5	73.86	75.64	73.96	74.79
LegalBERT	512	6	3	74.88	75.72	74.93	75.32
LegalBERT	512	6	5	75.02	75.67	75.05	75.36
RoBERTa	512	6	3	64.34	68.85	64.42	65.61
RoBERTa	512	6	5	69.08	70.11	69.14	69.62
LegalLED	1024	6	3	68.09	69.05	68.16	68.97
LED	1024	6	3	67.30	68.45	67.36	67.90

Table A.2: Complete results of obtained by training Transformers models on $ILDC_{multi}$. The "Tokens" column refers to the number of "last" tokens of each document exploited. The "E" column refers to the number of epochs. The columns "A", "P", and "R" refers to accuracy, precision, and recall, respectively.

Model	E1	E2	Pooling	A	P	R	F1
LegalBERT	3	2	[CLS]	77.42	77.44	77.44	77.43
LegalBERT	3	3	[CLS]	78.91	78.92	78.88	78.90
LegalBERT	3	5	[CLS]	76.87	76.88	76.85	76.87
LegalBERT	3	3	MEAN	77.57	77.61	77.39	77.50
LegalBERT	3	3	MAX	75.61	75.43	75.31	75.37
LegalBERT	5	3	[CLS]	75.47	75.60	75.55	75.57
LegalBERT	5	3	MEAN	76.41	76.45	76.33	76.39
LegalBERT	5	3	MAX	74.59	74.58	74.10	74.34
LegalLED	3	3	[CLS]	33.32	32.34	33.33	33.25
LegalLED	3	3	MEAN	66.41	66.54	66.47	66.45
LegalLED	3	3	MAX	64.88	64.94	64.34	64.00
LED	3	3	[CLS]	33.32	32.33	33.32	33.29
LED	3	3	MEAN	61.34	61.31	67.72	69.05
LED	3	3	MAX	59.18	55.01	57.02	62.74
RoBERTa	3	3	[CLS]	67.92	67.88	67.87	67.88
RoBERTa	3	3	MEAN	66.66	66.76	66.75	66.76
RoBERTa	3	3	MAX	68.50	68.56	67.91	68.24
RoBERTa	5	2	[CLS]	73.28	73.30	72.59	72.44
RoBERTa	5	3	[CLS]	72.68	72.78	72.33	72.55
RoBERTa	5	5	[CLS]	73.74	73.11	69.78	71.41
RoBERTa	5	3	MEAN	70.01	70.42	67.73	69.06
RoBERTa	5	3	MAX	66.67	67.74	58.43	62.74
CaseLawBERT	3	2	[CLS]	81.02	81.15	81.07	81.12
CaseLawBERT	3	2	MEAN	78.02	78.01	77.97	78.00
CaseLawBERT	3	3	MAX	74.09	72.66	73.65	73.68
CaseLawBERT	5	3	[CLS]	80.04	80.60	80.23	79.71
CaseLawBERT	5	3	MEAN	69.20	69.12	69.07	69.10
CaseLawBERT	5	3	MAX	77.02	76.61	75.89	76.25
LegalLSGBERT	3	3	[CLS]	82.06	82.07	82.06	82.13
LegalLSGBERT	3	3	MEAN	80.87	80.86	80.81	80.84
LegalLSGBERT	3	3	MAX	77.91	77.84	77.83	77.84

Table A.3: Complete results of obtained by training Hierarchical Transformers models on $ILDC_{multi}$. Note that the column "E1" refers to the number of epochs for which the transformer was trained for embedding extraction, while the column "E2" refers to the number of epochs for the aggregation model. The columns "A", "P", and "R" refers to accuracy, precision, and recall, respectively.

Metric (50% of sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.378	0.343	0.385	0.386	0.350
Overlap-Min	0.683	0.620	0.729	0.756	0.624
Overlap-Max	0.470	0.439	0.456	0.450	0.449
ROUGE-1	0.542	0.509	0.542	0.557	0.517
ROUGE-2	0.391	0.341	0.406	0.419	0.356
ROUGE-L	0.526	0.481	0.529	0.548	0.493
BLEU	0.274	0.334	0.218	0.214	0.334
METEOR	0.273	0.353	0.234	0.232	0.342

Table A.4: Explanation generation results using the Occlusions method with LegalLSGBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 50.

Metric (30% of sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.279	0.274	0.268	0.268	0.274
Overlap-Min	0.700	0.554	0.753	0.786	0.596
Overlap-Max	0.328	0.363	0.299	0.294	0.351
ROUGE-1	0.429	0.426	0.411	0.422	0.428
ROUGE-2	0.269	0.241	0.265	0.276	0.256
ROUGE-L	0.412	0.395	0.399	0.414	0.404
BLEU	0.092	0.194	0.046	0.041	0.170
METEOR	0.157	0.231	0.126	0.123	0.216

Table A.5: Explanation generation results using the Occlusions method with LegalLSGBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 30.

Metric (20% of sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.211	0.215	0.191	0.189	0.210
Overlap-Min	0.718	0.561	0.760	0.789	0.604
Overlap-Max	0.239	0.272	0.206	0.201	0.254
ROUGE-1	0.342	0.351	0.312	0.318	0.345
ROUGE-2	0.196	0.177	0.178	0.186	0.182
ROUGE-L	0.329	0.322	0.301	0.312	0.323
BLEU	0.028	0.087	0.009	0.006	0.065
METEOR	0.098	0.147	0.075	0.073	0.137

Table A.6: Explanation generation results using the Occlusions method with LegalLSGBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 20.

Metric (50% of sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.385	0.34	0.386	0.395	0.348
Overlap-Min	0.681	0.627	0.714	0.751	0.623
Overlap-Max	0.479	0.435	0.467	0.466	0.446
ROUGE-1	0.556	0.511	0.546	0.567	0.518
ROUGE-2	0.407	0.352	0.412	0.435	0.361
ROUGE-L	0.544	0.49	0.533	0.559	0.497
BLEU	0.305	0.34	0.266	0.263	0.338
METEOR	0.291	0.375	0.257	0.26	0.354

Table A.7: Explanation generation results using the Occlusions method with CaseLawBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 50.

Metric (30% of sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.299	0.292	0.271	0.27	0.277
Overlap-Min	0.714	0.561	0.747	0.778	0.576
Overlap-Max	0.349	0.388	0.303	0.297	0.359
ROUGE-1	0.455	0.448	0.415	0.425	0.431
ROUGE-2	0.299	0.276	0.272	0.284	0.258
ROUGE-L	0.443	0.423	0.404	0.418	0.406
BLEU	0.099	0.236	0.05	0.047	0.19
METEOR	0.167	0.255	0.131	0.131	0.224

Table A.8: Explanation generation results using the Occlusions method with CaseLawBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 30.

Metric (20% of sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.220	0.230	0.192	0.19	0.216
Overlap-Min	0.733	0.574	0.762	0.79	0.599
Overlap-Max	0.245	0.287	0.207	0.202	0.261
ROUGE-1	0.354	0.369	0.313	0.319	0.351
ROUGE-2	0.207	0.198	0.181	0.188	0.185
ROUGE-L	0.344	0.345	0.304	0.313	0.329
BLEU	0.026	0.103	0.008	0.007	0.072
METEOR	0.103	0.163	0.079	0.078	0.139

Table A.9: Explanation generation results using the Occlusions method with CaseLawBERT. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 20.

Metric (50% of the sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.407	0.392	0.429	0.429	0.396
Overlap-Min	0.727	0.656	0.797	0.822	0.661
Overlap-Max	0.490	0.498	0.487	0.479	0.503
ROUGE-1	0.584	0.567	0.605	0.618	0.575
ROUGE-2	0.449	0.417	0.492	0.501	0.433
ROUGE-L	0.568	0.541	0.595	0.611	0.554
BLEU	0.313	0.422	0.258	0.249	0.417
METEOR	0.300	0.411	0.263	0.256	0.389

Table A.10: Explanation generation results using the Occlusions method with LegalLSGBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 50.

Metric (30% of the sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.302	0.311	0.297	0.293	0.308
Overlap-Min	0.749	0.598	0.822	0.848	0.639
Overlap-Max	0.347	0.401	0.323	0.312	0.382
ROUGE-1	0.462	0.468	0.453	0.460	0.470
ROUGE-2	0.317	0.293	0.322	0.329	0.306
ROUGE-L	0.447	0.436	0.443	0.453	0.445
BLEU	0.114	0.244	0.059	0.05	0.198
METEOR	0.179	0.263	0.142	0.135	0.241

Table A.11: Explanation generation results using the Occlusions method with LegalLSGBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 30.

Metric (20% of the sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.235	0.257	0.214	0.213	0.244
Overlap-Min	0.772	0.626	0.828	0.866	0.664
Overlap-Max	0.262	0.311	0.226	0.222	0.287
ROUGE-1	0.375	0.398	0.345	0.355	0.385
ROUGE-2	0.237	0.227	0.220	0.231	0.224
ROUGE-L	0.365	0.370	0.337	0.351	0.362
BLEU	0.047	0.111	0.014	0.008	0.083
METEOR	0.117	0.177	0.087	0.083	0.158

Table A.12: Explanation generation results using the Occlusions method with LegalLSGBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 20.

Metric (50% of the sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.422	0.378	0.434	0.436	0.382
Overlap-Min	0.712	0.667	0.768	0.799	0.658
Overlap-Max	0.517	0.469	0.505	0.498	0.480
ROUGE-1	0.595	0.554	0.596	0.610	0.560
ROUGE-2	0.464	0.405	0.479	0.491	0.415
ROUGE-L	0.585	0.538	0.587	0.605	0.542
BLEU	0.35	0.380	0.300	0.292	0.377
METEOR	0.317	0.397	0.275	0.275	0.382

Table A.13: Explanation generation results using the Occlusions method with CaseLawBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 50.

Metric (30% of the sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.311	0.308	0.296	0.293	0.302
Overlap-Min	0.728	0.589	0.789	0.819	0.613
Overlap-Max	0.362	0.397	0.324	0.317	0.384
ROUGE-1	0.472	0.471	0.446	0.455	0.465
ROUGE-2	0.328	0.303	0.315	0.321	0.304
ROUGE-L	0.463	0.451	0.439	0.45	0.447
BLEU	0.119	0.247	0.065	0.063	0.216
METEOR	0.182	0.272	0.146	0.144	0.245

Table A.14: Explanation generation results using the Occlusions method with CaseLawBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 30.

Metric (20% of the sentences)	Explainability Model vs Expert				
	Expert				
	1	2	3	4	5
Jaccard Similarity	0.235	0.243	0.209	0.212	0.239
Overlap-Min	0.754	0.589	0.792	0.84	0.628
Overlap-Max	0.261	0.301	0.223	0.223	0.284
ROUGE-1	0.374	0.388	0.336	0.349	0.383
ROUGE-2	0.235	0.223	0.212	0.225	0.228
ROUGE-L	0.365	0.369	0.329	0.346	0.366
BLEU	0.037	0.115	0.01	0.011	0.085
METEOR	0.115	0.177	0.086	0.088	0.153

Table A.15: Explanation generation results using the Occlusions method with CaseLawBERT and the revised version of the $ILDC_{expert}$ corpus. The "Mean" column represents the average scores computed among the five legal experts. The percentage of the best sentences extracted is equal to 20.

Bibliography

[1] Paulo Quaresma and Teresa Gonçalves. «Using Linguistic Information and Machine Learning Techniques to Identify Entities from Juridical Documents». In: *Semantic Processing of Legal Texts: Where the Language of Law Meets the Law of Language*. Ed. by Enrico Francesconi, Simonetta Montemagni, Wim Peters, and Daniela Tiscornia. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 44–59. ISBN: 978-3-642-12837-0. DOI: 10.1007/978-3-642-12837-0_3. URL: https://doi.org/10.1007/978-3-642-12837-0_3 (cit. on p. 1).

[2] Marc Queudot, Éric Charton, and Marie-Jean Meurs. «Improving Access to Justice with Legal Chatbots». In: *Stats* 3.3 (2020), pp. 356–375. ISSN: 2571-905X. DOI: 10.3390/stats3030023. URL: https://www.mdpi.com/2571-905X/3/3/23 (cit. on p. 1).

[3] Marina Valpeters, Ivan Kireev, and Nikolay Ivanov. «Application of machine learning methods in big data analytics at management of contracts in the construction industry». In: *MATEC Web of Conferences*. Vol. 170. EDP Sciences. 2018, p. 01106 (cit. on p. 1).

[4] Deepali Jain, Malaya Dutta Borah, and Anupam Biswas. «Summarization of legal documents: Where are we now and the way forward». In: *Computer Science Review* 40 (2021), p. 100388 (cit. on p. 1).

[5] Vijit Malik, Rishabh Sanjay, Shubham Kumar Nigam, Kripa Ghosh, Shouvik Kumar Guha, Arnab Bhattacharya, and Ashutosh Modi. *ILDC for CJPE: Indian Legal Documents Corpus for Court Judgment Prediction and Explanation*. 2021. DOI: 10.48550/ARXIV.2105.13562. URL: https://arxiv.org/abs/2105.13562 (cit. on pp. 2, 24, 31–34, 36–38, 40, 41, 47–49, 51, 54, 58–60, 69).

[6] Shohreh Shaghaghian, Luna Feng, B. Jafarpour, and Nicolai Pogrebnyakov. «Customizing Contextualized Language Models for Legal Document Reviews». In: *2020 IEEE International Conference on Big Data (Big Data)* (2020), pp. 2139–2148. DOI: 10.1109/BigData50022.2020.9378201 (cit. on p. 2).

[7] Neel Guha et al. *LegalBench: A Collaboratively Built Benchmark for Measuring Legal Reasoning in Large Language Models*. 2023. arXiv: 2308.11462 [cs.CL] (cit. on p. 2).

[8] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. 2016. arXiv: 1602.04938 [cs.LG] (cit. on pp. 2, 32).

[9] Kacper Sokol and Peter A. Flach. «Conversational Explanations of Machine Learning Predictions Through Class-contrastive Counterfactual Statements». In: (2018), pp. 5785–5786. DOI: 10.24963/ijcai.2018/836 (cit. on p. 2).

[10] Li Deng and Yang Liu. *Deep Learning in Natural Language Processing*. 1st. Springer Publishing Company, Incorporated, 2018. ISBN: 9789811052088 (cit. on p. 4).

[11] Ali Darwish. «Linguistic and Epistemic Inference in Cross-Cultural Communication in Satellite Television News Media». In: (Mar. 2023) (cit. on p. 5).

[12] Juan Ramos et al. «Using tf-idf to determine word relevance in document queries». In: *Proceedings of the first instructional conference on machine learning*. Vol. 242. 1. Citeseer. 2003, pp. 29–48 (cit. on p. 7).

[13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. arXiv: 1301.3781 [cs.CL] (cit. on pp. 7, 10).

[14] Quoc V. Le and Tomás Mikolov. «Distributed Representations of Sentences and Documents». In: *CoRR* abs/1405.4053 (2014). arXiv: 1405.4053. URL: http://arxiv.org/abs/1405.4053 (cit. on pp. 8–10).

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. *Distributed Representations of Words and Phrases and their Compositionality*. 2013. arXiv: 1310.4546 [cs.CL] (cit. on p. 9).

[16] Quoc V. Le and Tomas Mikolov. *Distributed Representations of Sentences and Documents*. 2014. arXiv: 1405.4053 [cs.CL] (cit. on p. 9).

[17] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. «Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features». In: *CoRR* abs/1703.02507 (2017). arXiv: 1703.02507. URL: http://arxiv.org/abs/1703.02507 (cit. on p. 11).

[18] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL] (cit. on p. 11).

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. «Attention Is All You Need». In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762 (cit. on pp. 13, 14).

[20] Tom B. Brown et al. «Language Models are Few-Shot Learners». In: *CoRR* abs/2005.14165 (2020). arXiv: 2005.14165. URL: https://arxiv.org/abs/2005.14165 (cit. on p. 13).

[21] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805 (cit. on pp. 13–16, 31).

[22] Yinhan Liu et al. «RoBERTa: A Robustly Optimized BERT Pretraining Approach». In: *CoRR* abs/1907.11692 (2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692 (cit. on pp. 15, 16).

[23] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. «XLNet: Generalized Autoregressive Pretraining for Language Understanding». In: *CoRR* abs/1906.08237 (2019). arXiv: 1906.08237. URL: http://arxiv.org/abs/1906.08237 (cit. on p. 16).

[24] Iz Beltagy, Matthew E. Peters, and Arman Cohan. *Longformer: The Long-Document Transformer*. 2020. arXiv: 2004.05150 [cs.CL] (cit. on pp. 17, 18, 20).

[25] Manzil Zaheer et al. «Big Bird: Transformers for Longer Sequences». In: *CoRR* abs/2007.14062 (2020). arXiv: 2007.14062. URL: https://arxiv.org/abs/2007.14062 (cit. on pp. 19, 20, 65).

[26] Charles Condevaux and Sébastien Harispe. *LSG Attention: Extrapolation of pretrained Transformers to long sequences*. 2022. arXiv: 2210.15497 [cs.CL] (cit. on pp. 20, 21).

[27] Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. «Efficient Attentions for Long Document Summarization». In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou. Online: Association for Computational Linguistics, June 2021, pp. 1419–1436. DOI: 10.18653/v1/2021.naacl-main.112. URL: https://aclanthology.org/2021.naacl-main.112 (cit. on p. 20).

[28] Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya Razenshteyn, and Ludwig Schmidt. *Practical and Optimal LSH for Angular Distance*. 2015. arXiv: 1509.02897 [cs.DS] (cit. on p. 20).

[29] Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. «LEGAL-BERT: The Muppets straight out of Law School». In: *CoRR* abs/2010.02559 (2020). arXiv: 2010.02559. URL: https://arxiv.org/abs/2010.02559 (cit. on pp. 28, 29).

[30] Lucia Zheng, Neel Guha, Brandon R. Anderson, Peter Henderson, and Daniel E. Ho. «When Does Pretraining Help? Assessing Self-Supervised Learning for Law and the CaseHOLD Dataset». In: *CoRR* abs/2104.08671 (2021). arXiv: 2104.08671. URL: https://arxiv.org/abs/2104.08671 (cit. on p. 29).

[31] Taku Kudo and John Richardson. «SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing». In: *CoRR* abs/1808.06226 (2018). arXiv: 1808.06226. URL: http://arxiv.org/abs/1808.06226 (cit. on p. 29).

[32] Fred Kort. «Predicting Supreme Court Decisions Mathematically: A Quantitative Analysis of the "Right to Counsel" Cases». In: *American Political Science Review* 51.1 (1957), pp. 1–12. DOI: 10.2307/1951767 (cit. on p. 30).

[33] S. Nagel. «Predicting court cases quantitatively». In: *Michigan Law Review* (1965) (cit. on p. 30).

[34] BENJAMIN E. LAUDERDALE and TOM S. CLARK. «The Supreme Court's Many Median Justices». In: *American Political Science Review* 106.4 (2012), pp. 847–866. DOI: 10.1017/S0003055412000469 (cit. on p. 30).

[35] R. Keown. «Mathematical Models For Legal Prediction». In: *UIC John Marshall Journal of Information Technology Privacy Law* (1980) (cit. on p. 30).

[36] Wan-Chen Lin, Tsung-Ting Kuo, Tung-Jia Chang, Chueh-An Yen, Chao-Ju Chen, and Shou-de Lin. « (Exploiting Machine Learning Models for Chinese Legal Documents Labeling, Case Classification, and Sentencing Prediction) [In Chinese]». In: *International Journal of Computational Linguistics & Chinese Language Processing, Volume 17, Number 4, December 2012-Special Issue on Selected Papers from ROCLING XXIV*. Ed. by Liang-Chih Yu, Richard Tzong-Han Tsai, Chia-Ping Chen, Cheng-Zen Yang, and Shu-Kai Hsieh. Dec. 2012. URL: https://aclanthology.org/O12-5004 (cit. on p. 30).

[37] Nikolaos Aletras, Dimitrios Tsarapatsanis, Daniel Preoţiuc-Pietro, and Vasileios Lampos. «Predicting judicial decisions of the European Court of Human Rights: a Natural Language Processing perspective». In: *PeerJ Computer Science* 2 (2016), e93. DOI: 10.7717/peerj-cs.93 (cit. on p. 30).

[38] Octavia-Maria Sulea, Marcos Zampieri, Shervin Malmasi, Mihaela Vela, Liviu P. Dinu, and Josef van Genabith. «Exploring the Use of Text Classification in the Legal Domain». In: *CoRR* abs/1710.09306 (2017). arXiv: 1710.09306. URL: http://arxiv.org/abs/1710.09306 (cit. on p. 30).

[39] Bingfeng Luo, Yansong Feng, Jianbo Xu, Xiang Zhang, and Dongyan Zhao. «Learning to Predict Charges for Criminal Cases with Legal Basis». In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Ed. by Martha Palmer, Rebecca Hwa, and Sebastian Riedel. Copenhagen, Denmark: Association for Computational Linguistics, Sept. 2017, pp. 2727–2736. DOI: 10.18653/v1/D17-1289. URL: https://aclanthology.org/D17-1289 (cit. on p. 30).

[40] Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Chaojun Xiao, Zhiyuan Liu, and Maosong Sun. «Legal Judgment Prediction via Topological Learning». In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Ed. by Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 3540–3549. DOI: 10.18653/v1/D18-1390. URL: https://aclanthology.org/D18-1390 (cit. on p. 31).

[41] Zikun Hu, Xiang Li, Cunchao Tu, Zhiyuan Liu, and Maosong Sun. «Few-Shot Charge Prediction with Discriminative Legal Attributes». In: *Proceedings of the 27th International Conference on Computational Linguistics*. Ed. by Emily M. Bender, Leon Derczynski, and Pierre Isabelle. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 487–498. URL: https://aclanthology.org/C18-1041 (cit. on p. 31).

[42] Nuo Xu, Pinghui Wang, Long Chen, Li Pan, Xiaoyan Wang, and Junzhou Zhao. *Distinguish Confusing Law Articles for Legal Judgment Prediction*. 2020. arXiv: 2004.02557 [cs.CL] (cit. on p. 31).

[43] Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. *Neural Legal Judgment Prediction in English*. 2019. arXiv: 1906.02059 [cs.CL] (cit. on pp. 31, 34).

[44] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. «BioBERT: a pre-trained biomedical language representation model for biomedical text mining». In: *CoRR* abs/1901.08746 (2019). arXiv: 1901.08746. URL: http://arxiv.org/abs/1901.08746 (cit. on pp. 31, 49).

[45] Iz Beltagy, Kyle Lo, and Arman Cohan. «SciBERT: A Pretrained Language Model for Scientific Text». In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Ed. by Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3615–3620. DOI: 10.18653/v1/D19-1371. URL: https://aclanthology.org/D19-1371 (cit. on pp. 31, 49).

[46] Scott Lundberg and Su-In Lee. *A Unified Approach to Interpreting Model Predictions*. 2017. arXiv: 1705.07874 [cs.AI] (cit. on p. 32).

[47] Hai Ye, Xin Jiang, Zhunchen Luo, and Wenhan 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)*. Ed. by Marilyn Walker, Heng Ji, and Amanda Stent. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 1854–1864. DOI: 10.18653/v1/N18-1168. URL: https://aclanthology.org/N18-1168 (cit. on p. 33).

[48] Xin Jiang, Hai Ye, Zhunchen Luo, WenHan Chao, and Wenjia Ma. «Interpretable Rationale Augmented Charge Prediction System». In: *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*. Ed. by Dongyan Zhao. Santa Fe, New Mexico: Association for Computational Linguistics, Aug. 2018, pp. 146–151. URL: https://aclanthology.org/C18-2032 (cit. on p. 33).

[49] Haoxi Zhong, Yuzhong Wang, Cunchao Tu, Tianyang Zhang, Zhiyuan Liu, and Maosong Sun. «Iteratively Questioning and Answering for Interpretable Legal Judgment Prediction». In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 2020, pp. 1250–1257. DOI: 10.1609/AAAI.V34I01.5479. URL: https://doi.org/10.1609/aaai.v34i01.5479 (cit. on p. 33).

[50] Matthew D. Zeiler and Rob Fergus. «Visualizing and Understanding Convolutional Networks». In: *Computer Vision – ECCV 2014*. Ed. by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars. Cham: Springer International Publishing, 2014, pp. 818–833. ISBN: 978-3-319-10590-1 (cit. on p. 33).

[51] Jiwei Li, Will Monroe, and Dan Jurafsky. «Understanding Neural Networks through Representation Erasure». In: *CoRR* abs/1612.08220 (2016). arXiv: 1612.08220. URL: http://arxiv.org/abs/1612.08220 (cit. on p. 33).

[52] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. arXiv: 1908.10084 [cs.CL] (cit. on p. 35).

[53] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. «Hierarchical Attention Networks for Document Classification». In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Kevin Knight, Ani Nenkova, and Owen Rambow. San Diego, California: Association for Computational Linguistics, June 2016, pp. 1480–1489. DOI: 10.18653/v1/N16-1174. URL: https://aclanthology.org/N16-1174 (cit. on p. 37).

[54] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG] (cit. on p. 42).

[55] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG] (cit. on p. 42).

[56] Daniel S. Wilks. «On the Combination of Forecast Probabilities for Consecutive Precipitation Periods». In: *Weather and Forecasting* 5.4 (1990), pp. 640–650. DOI: https://doi.org/10.1175/1520-0434(1990)005<0640:OTCOFP>2.0.CO;2. URL: https://journals.ametsoc.org/view/journals/wefo/5/4/1520-0434_1990_005_0640_otcofp_2_0_co_2.xml (cit. on p. 45).

[57] Peter Xenopoulos, Joao Rulff, Luis Gustavo Nonato, Brian Barr, and Claudio Silva. *Calibrate: Interactive Analysis of Probabilistic Model Output*. 2022. arXiv: 2207.13770 [cs.HC] (cit. on p. 45).

[58] Chin-Yew Lin. «ROUGE: A Package for Automatic Evaluation of Summaries». In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 74–81. URL: https://aclanthology.org/W04-1013 (cit. on p. 46).

[59] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. «Bleu: a Method for Automatic Evaluation of Machine Translation». In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Ed. by Pierre Isabelle, Eugene Charniak, and Dekang Lin. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics, July 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://aclanthology.org/P02-1040 (cit. on p. 46).

[60] Satanjeev Banerjee and Alon Lavie. «METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments». In: *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Ed. by Jade Goldstein, Alon Lavie, Chin-Yew Lin, and Clare Voss. Ann Arbor, Michigan: Association for Computational Linguistics, June 2005, pp. 65–72. URL: https://aclanthology.org/W05-0909 (cit. on p. 47).

[61] Ilias Chalkidis and Anders Søgaard. *Improved Multi-label Classification under Temporal Concept Drift: Rethinking Group-Robust Algorithms in a Label-Wise Setting*. 2022. arXiv: 2203.07856 [cs.CL] (cit. on p. 69).