UNIVERSITÀ DI PISA

Human Language Technology Project Report

Legal Judgement Predictor

Usage of LLMs embeddings in classification of legal cases from ECHR dataset

Students:

Giacomo Lagomarsini - mat. 583374
Leonardo Stoppani - mat. 580486

Lecturers:

Prof. Giuseppe Attardi

MSc in Computer Science curriculum AI

A.Y. 2022/2023

Abstract

Efficiently processing long documents for classification tasks remains a challenge in natural language processing. In this report, we address this challenge by using a chunk-based hierarchical transformer. This method involves segmenting documents into manageable chunks, employing a LLMs model to encode each chunk individually and a self-attention mechanism to capture the relationship between them. We explore various classification models, including Random Forest, Support Vector Machine (SVM), LSTM, and attention networks, on both single and multiple chunks. Additionally, we introduce sparse transformers, inspired by recent advancements in attention mechanisms, such as LongFormer and BigBird, to handle longer sequences. We evaluate our models on a standard dataset and compare their performance with existing methods. Results demonstrate the effectiveness of the proposed Hierarchical Bert approach, achieving competitive accuracy, precision, recall, and F1 scores while efficiently handling long documents. Moreover, our implementation achieves comparable results to state-of-the-art models like HierBert. Our findings suggest that chunk-based hierarchical transformers offer a promising solution for efficient text classification tasks, particularly for processing lengthy documents.

Contents

1 Introduction

The fusion of advanced natural language processing (NLP) techniques with legal studies represents a new exciting frontier within the field of Human Language Technology. This project specifically focuses on *European Court of Human Rights (ECHR)* cases, which present a rich and complex dataset for analysis [1]. We worked on the binary classification task that is, given the facts of a case, we want to classify as positive if any human rights article or protocol has been violated, negative otherwise.

There are several challenging aspects to this task. Firstly, the intersection of the legal domain with human language technologies poses relevant problems related to ethics, results reliability, and model explainability. Moreover, it is necessary to mitigate potential biases in both the legal and NLP domains.

From a purely technical point of view, legal documents are notably longer compared to typical texts used in NLP tasks such as sentiment analysis. We will analyze this in Section 3. Practically, this reflects in standard language models used for text classification, like *BERT* [2], having a maximum number of input tokens way smaller than our documents. The limitation on context length in models is not arbitrary. For a sequence of length l, a typical transformer architecture requires storing $\mathcal{O}(l^2)$ self-attention coefficients, corresponding to each pair of tokens in the sequence, for each layer. This characteristic makes them less suitable for processing long contexts, particularly in scenarios with limited or shared memory resources, such as the machine used for this project.

Therefore, our primary contribution in this report revolves around exploring methods to efficiently handle long sequences, addressing challenges posed by the memory constraints inherent in large-scale transformer architectures.

The report first introduces the related works in Section 2, then describes in details the dataset object of our studies in Section 3. Further, the report dives into the experiment that we conducted, in Section 4 it explains how we manage to get Bert embeddings and our first approach to classification, while in Section 6 it presents our implementation of a Hierarchical Transformer. In Section 5 we briefly explain our experiments with Sparse Transformers. Finally in Section 7 we show the experiment result with the Hierarchical Transformer model outperforming the others and in Section 8 we discuss how and in which direction our work could be developed further in the future.

2 Related Works

The dataset we used was originally published by Chalkidis et al. [3] in their work. In this study, various methods were explored to address the challenge posed by long sequences. The concept of hierarchical transformers was introduced, demonstrating that, with its limited context, BERT struggled to accurately classify cases. Our work differs from the one of Chalkidis et al. in the analysis of the embeddings and

number of chunks. Compared to them we also achieved promising results by fixing the context length of our models to 512 tokens.

Another study, from Medvedeva et al. [4] provide an extensive review that offers a panoramic view of classification tasks related to legal datasets. They introduce a new perspective on the legal cases classification as distinct from forecasting, which involves predicting the outcome of ongoing legal proceedings. They argue that while forecasting involves predicting the final judgement of ongoing legal proceedings based on available evidence, legal outcome classification focuses on categorizing past cases into predefined sentences. This differentiation is crucial as it affects the methodology and interpretation of results in legal NLP tasks. Moreover, Medvedeva et al. emphasize the significance of considering the societal and contextual factors surrounding legal cases, as these can influence the interpretation and application of legal texts. Finally, they state that describing the details and facts of a legal case post-verdict inherently introduces bias.

Furthermore, our task is part of the LexGLUE dataset [5], which aims to serve as a benchmark for natural language understanding in the legal domain. LexGLUE encompasses a range of NLP tasks specifically designed to legal text comprehension. This dataset facilitates the evaluation and comparison of different models and techniques within the legal context, enabling advancements in the field of Human Language Technology applied to legal studies.

3 Dataset Analysis

The dataset used in our study is sourced from the European Court of Human Rights (ECHR), which adjudicates allegations of human rights violations within the framework of the European Convention on Human Rights 1. The dataset, comprising approximately 11.5k cases sourced from the ECHR's public database, provides a comprehensive repository of factual descriptions extracted from case documents using regular expressions, akin to the methodology outlined in [6]. Each case is tagged with the articles of the Convention that were allegedly violated, if applicable. Moreover, the dataset includes an importance score assigned by the ECHR, offering additional insights into the significance of each case. As illustrated in Figure 1, the mean number of tokens in the ECHR dataset is nearly 3000, while standard BERT-based models, for instance, can only process up to 512 tokens at a time.

Subset	**Train**	**Dev.**	**Test**
Cases	7100	1380	2998

Table 1: Subsets of the ECHR dataset partition.

To facilitate model training and evaluation, the dataset is partitioned into distinct training, development, and test sets, as described in Table 1. The training and

```
{
    "ITEMID": "001-72594",
    "LANGUAGEISOCODE": "ENG",
    "RESPONDENT": "UKR",
    "BRANCH": "CHAMBER",
    "DATE": 2006,
    "DOCNAME": "CASE OF SHCHUKIN v. UKRAINE",
    "IMPORTANCE": "4",
    "CONCLUSION": "Violation of Art. 6-1;Not necessary to examine
    ↪  Art. 13",
    "TEXT": [
        "4. The applicant was born in 1940 and lives in the village
        ↪  of Veseloye, Kherson region, Ukraine.",
        "5. In 2001 the applicant instituted proceedings in the
        ↪  Novokakhovskiy Town Court against his former employer,
        ↪  the State-owned \u201cUzhelektromash\u201d company, to
        ↪  recover salary arrears. On 17 December 2001 the court
        ↪  awarded the applicant UAH 10,282 in salary arrears and
        ↪  compensation.",
    ],
    "VIOLATED_ARTICLES": [
        "6"
}
```


Listing 1: ECHR dataset entry example in json format.

development sets span cases from 1959 to 2013, while the test set covers cases from 2014 to 2018. The training and development sets are balanced, containing an equal distribution of cases with and without violations. The balancing should ensure that the models trained on these sets remain unbiased towards any specific class, thereby enhancing the generalization capability of the results.

In contrast, the test set comprises a higher proportion (66%) of cases with violations, mirroring the approximate ratio observed in the database. This skew toward cases with violations in the test set aims to more accurately reflect real-world scenarios. Furthermore, the test set presents a challenge for models due to the presence of labels not encountered during training, with 45 out of 66 labels absent from the training set and an additional 11 labels represented in fewer than 50 cases. This characteristic makes the dataset particularly suitable for evaluating the performance of models under few-shot learning scenarios, where models must generalize effectively

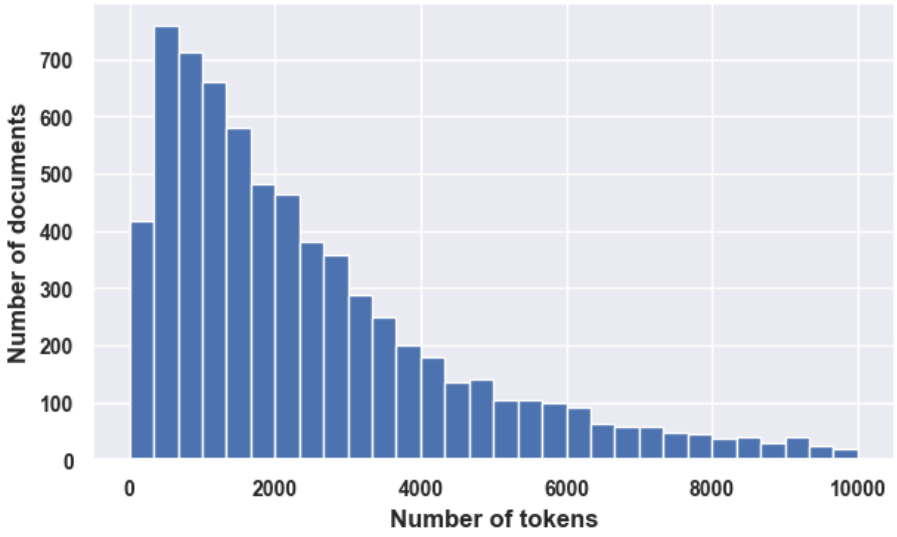
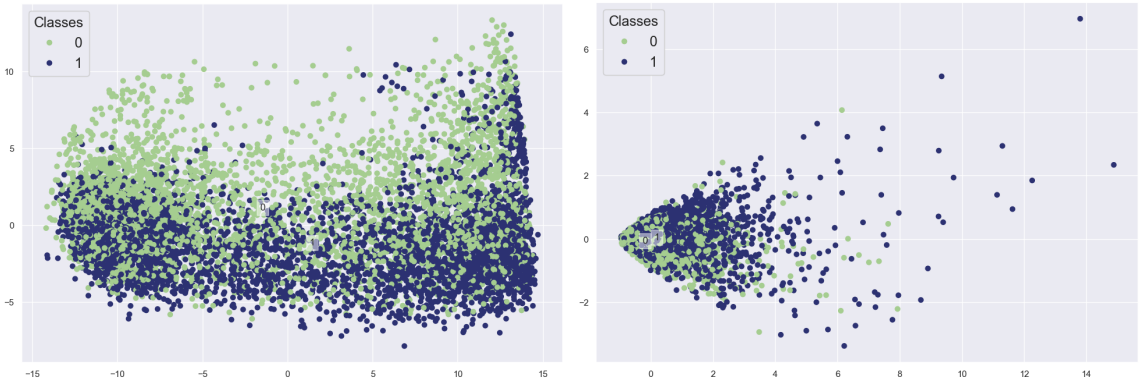


Figure 1: Distribution of the number of tokens across all the documents

to novel classes with limited training data.

We focused on a binary classification task violated/non violated, where we classify as positive a case where at least one article was violated. This is sensible because in the ECHR jurisdiction, the allegedly violated article are presented at the beginning of the case, as noticed in [4]. In the next sections we explore different approaches we have taken to tackle the task.

4 Bert Embeddings



(a) First embedding chunk.

(b) Average embedding chunks.

Figure 2: Principal Component Analysis (PCA) of the training set.

The first approach we took was trying to classify the embedding given by a large

language model. This is a very lightweight approach, and can serve as a solid baseline.

From all the models available in the *Huggingface library* [7], we opted for Legal BERT [8]. This version of BERT is further pretrained on a large corpus of legal documents and we found it ideal for our task. In particular, the very same dataset we take in consideration was used for the masked language pretraining of the model.

4.1 Chunks

Because of the limited number of tokens that Bert (and its finetuned version) takes in input, we need to split the documents in chunks. We use a dynamic approach for tokenizing and splitting in chunks the documents. Let $d = \{s_1, ..., s_n\}$ be a document and $d_{tok} = \{t_1, ..., t_n\}$ be the tokenized document. Let n_i be the length in tokens of tokenized sentence t_i. Then, we split the tokenized document in chunks in this way:

$$C_1 = [t_1^{(1)}, \ldots, t_{m_1}^{(1)}, \text{ padding}], \ldots, C_k = [t_1^{(k)}, \ldots, t_{m_k}^{(k)}, \text{ padding}]$$

Where for each chunks C_j, $t_1^{(j)}, ..., t_{m_j}^{(j)}$ are such that their total length $\sum_{i=1}^{m_j} n_i$ does not exceed 512 (the maximum number of tokens taken in input by Bert), but adding the next tokenized sentence $t_{m_{j+1}}^{(j)}$ would result in the chunk's length exceeding 512. The padding is added so that all the chunks have the same length of 512, for efficiency reasons. Doing this, we aim to minimizing the padding needed, while not truncating any sentence in the document.

As preliminary investigation we did a Principal Component Analysis of BERT embeddings, reported in Figure 2, which shows two components representing our classes.

4.2 Single Chunk Classification

Once we have the tokens, we can pass it just one time in our model and produce the embeddings, thus using the embedding for classification. Notice that, because

Model	First Chunk F1	Mean Chunks F1
Random Forest	0.70	0.65
SVM	0.76	0.70

Table 2: F1 score of Random Forest and SVM models both with the first chunk of embeddings and the average.

we use the entirety of each document, the output of the model given a document d_i will be $[E_{i,1}, ..., E_{i,n_i}]$ where E_j is the embedding of the [CLS] token of chunk C_j.

Because n_i depends on the document length, we had to find a way to handle the different lengths. The first, simplest, approach we took was using just the first embdedding $E_{i,1}$, or alternatively using the mean of the embedding $\bar{E}_i = \frac{1}{n_i} \sum E_{i,j}$.

We trained a Random Forest and a Support Vector Machine on both the first and the mean embedding. Interestingly, we noticed how using only the first embedding improved the performances of the models, with respect to using the average embedding, as we shown in Table 2. This could mean that most of the relevant information is in the first few sentences of the documents. As we also noticed from the PCA, a standard average of the embeddings seems to introduce noise in the principal components. Nevertheless, with a smarter approach, we also tried to use the following chunks, as described in the next Section 4.3.

4.3 Multiple Chunks Classification

Model	Hyperparameters
Random Forest	`n_extimators` =100
SVM	`kernel` = linear
LSTM	`recurrent_size` =768, `feedforward_size` =16, `lr`=10^{-3},`weight_decay`=10^{-4}
Attention Network	`hidden_size`=16, `lr`=10^{-4}, `weight_decay`=10^{-3}

Table 3: Hyperparameters of the models in Section 4

The first model we tried is a simple LSTM, with a feedforward layer used for classification.

We also used an attention model that follows the approach used in Hierarchical Attention Networks [9]: given the values and keys $V_i = W_V E_i$ and $K_i = W_K E_i$, the query is a single vector q, that serves as a selector of the most meaningful chunks.

The output

$$X = \sum \alpha_i V_i \quad \text{with} \quad \alpha_i = softmax(e_i), \quad e_i = q^T K_i$$

is effectively a normalized weighted mean of the values produced from the embeddings. This produces a hierarchy in the attention, because first the part of the document are processed one by one and the last attention layer summarizes the whole text. For all the models, we performed a 5-fold cross validation grid search to find the best hyperparameters. Table 3 shows the best parameters we found.

The output X is then passed through a feedforward network and classified. As shown in Table 4, the recurrent neural network actually fails in utilizing all the chunks properly, and overall does worse than the models that utilize one single chunk. However, with the attention head the results are better of 2 points in F1 than the SVM.

This result motivates our next steps, where we try to extend the idea of hierarchical attention, by creating a hierarchical transformer that uses attention for the words in a single chunk, and a layer of cross-chunk attention.

Model	**Accuracy**	**Precision** (macro)	**recall** (macro)	**F1** (macro)
Random Forest	0.75	0.72	0.70	0.70
SVM	0.80	0.79	**0.75**	0.76
LSTM	0.74	0.72	0.68	0.70
Attention Network	**0.81**	**0.82**	**0.75**	**0.78**

Table 4: Test results of the models in Section 4. Random Forest and SVM use only one chunk of the embeddings, while LSTM and Attention Network use all the chunks.

5 Sparse Transformers

As already mentioned, one of the main problems of using vanilla self-attention for large sequences is the impact on memory: for every pair of tokens i, j it is necessary to compute an attention value α_{ij}. This reflects in a number of attention weights proportional to the square of the sequence length, and the time complexity to compute the values is also quadratic. This price has to be paid if we want to model the direct connection to any pair of words in our text: a cheaper approach like using sliding windows has the obvious disadvantage of not being able to capture any relationship outside of the window size. Therefore, various approaches have been used to have a compromise between the expressive power of attention and the cheapness of sliding windows. One approach can be having just a small subset of special tokens with a global attention, and the other tokens with a windowed attention. This way, we still retain part of the global information, while decreasing the number of tokens by an order of magnitude. This is the approach followed in LongFormer, by Beltagi et al. [10].

Random attention has also been proved useful in situation where efficiency counts: in this approach the pair i, j for which we compute α_{ij} are chosen at random. BigBird by Google DeepMind [11] combines the two approaches in one model, by having some tokens with global attention, the majority of the with local, window attention, and also random attention. We chose a pretrained instance of BigBird to test sparse attention for our task. The pretrained model is warm-started from RoBerta weights, i.e. its weigths are initialized from a pre-trained RoBerta model [11]. We fine-tuned the model for just one epoch, with a starting learning rate of `3e-6` and 0.01 of weight decay.

The maximum context length of the model is 4096 tokens, eight times more than Bert, and more than the median length of our texts. Some texts still have to be truncated, but we think that the number of tokens is sufficient for the model to reach satisfying results.

6 Hierarchical Transformers

To address the challenge posed by processing long sequences, we also have implemented a version of Hierarchical Transformers. The concept of employing a hierarchy of attention networks was initially introduced by Yang et al. (2016) [9], where they utilized various recurrent networks to encode different parts of documents at varying levels of granularity such as words, sentences, and paragraphs. This approach has since been further developed using large language models, as seen in works like Chalkidis et al. (2019) [3] and Chalkidis et al. (2022) [12]. The model architecture is straightforward: an encoder model of choice is employed to encode data into chunks and generate embeddings $E_1, .., E_n$, from the special token `[CLS]`. The encoder model is shared between the chunks, allowing the model to scale for large values of n. Due to memory constraints, the number of chunks (and hence embeddings) is limited and fixed to a positive integer n. In case a document has less than n chunks we employ a padding strategy. These chunks then undergo a self-attention mechanism to capture relationships between them, resulting in vectors $att(E_1), ..., att(E_n)$. These vectors are subsequently pooled (we used an average pooling) and fed into a feedforward layer with sigmoidal activation for final binary classification. The architecture of the model is depicted in Figure 3.

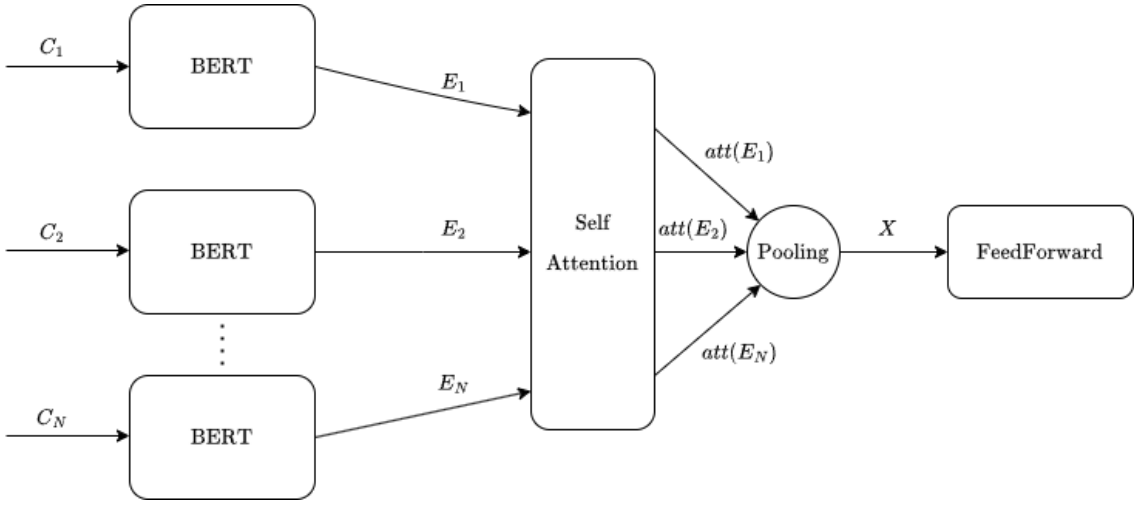


Figure 3: Architecture of Hierarchical Bert. The N BERT model share their parameters, allowing the model to scale.

This architecture has an interesting parallelism with models with sparse or local attention, that have been used in the literature to address long sequences. Model as LongFormer or BigBird have been proved to be successful in a variety of tasks pertaining long sequences and dependencies. In contrast to sparse attention networks, the attention relationships can be visualized as a blockwise diagonal matrix, akin to what is illustrated in Figure 4. Each block corresponds to a chunk computed inde-

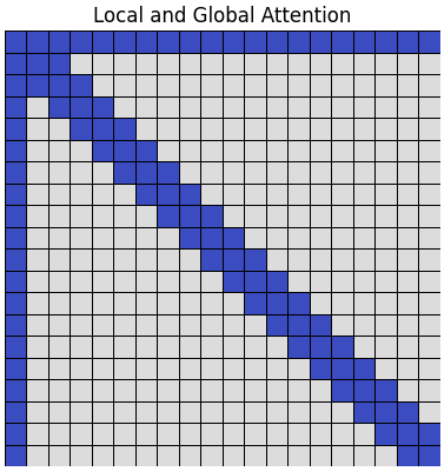
pendently. Furthermore, an additional step of cross-chunk self-attention is performed to enhance contextualization across the chunks.

6.1 Implementation details

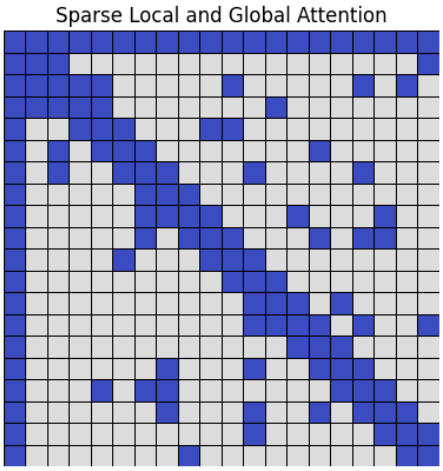
To train the model, we used the *Accelerate* [13] library provided by Huggingface, that allows data parallelism across the multiple GPUs provided for the project. Doing this, we can exploit fully the machine,without overloading a single GPU.

We used the Adam optimizer with a learning rate of `3e-6`. We also tried varying the number of chunks to give to the model. Obviously, with a lower number of chunks there is a huge gain in both memory usage and speed of training, because for each batch, we have to do a number of forward of the BERT part of the model equal to the number of chunks (while the backward pass is just one).

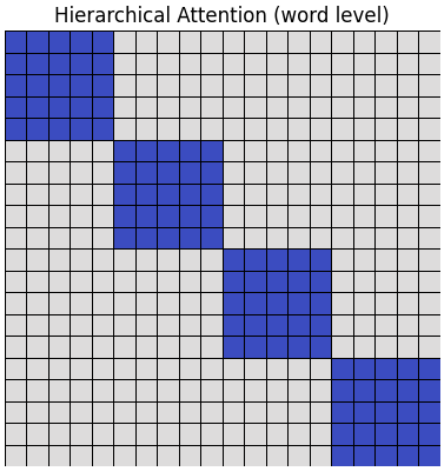
For memory reasons, the maximum number of chunks tried is 8, corresponding to a maximum of 4096 tokens, the same as BigBird. Concerning the memory and speed of training, training for four epochs Hierarchical Bert taking 8 chunks on 3 GPUs took a little less than 2 hours, just as long as the training of BigBird for just one epoch on the same data. This means that our model is approximately $4\times$ faster than the sparse transformer.



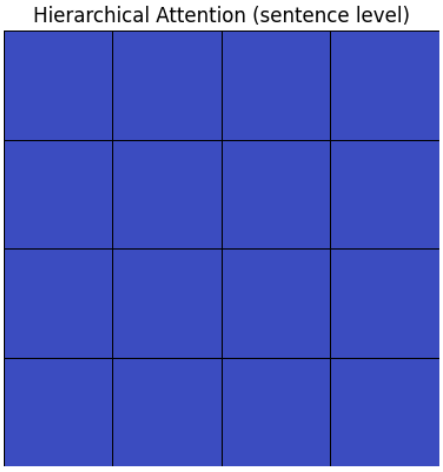
(a) Local and global attention, as in LongFormer



(b) Local, global and random attention as in BigBird



(c) Hierarchical attention as in HierBert



(d) Full attention on the chunks/sentences embeddings

Figure 4: Depiction of attention in long sequence transformers as Longformer and BigBird and in our approach. Longformer uses a mix of local (each token sees only its n nearest neighbors) and global attention. BigBird also adds random sparse attention coefficients. Hierarchical attention works in two phases: the document is first split in sentences (or groups of sentences), and the attention matrix becomes block-diagonal. The second phase involves a full attention on the sentences embeddings.

7 Results

All the models were trained, validated and tested on the same split, given by the authors of [3], who published the dataset. Before running the test, we choose the best hyperparameters for each model through a random grid-search. A k-fold cross validation was used to validate the model on the development set. We compared our three main approaches:

- BigBird, a transformer with enhanced context length that uses sparse and local attention as well as global attention (Section 5).
- A fine-tuned version of Bert, with 512 token max length.
- Hierarchical Bert with 2, 4 and 8 chunks, corresponding to 1024, 2048, 4096 tokens respectively (Section 6).

We also compared our results with the one in [3]. They do not specify the number of chunks used in the HierBert approach, so we assume that they managed to encode all the documents in their entirety. Notably, we found good results with a simple fine-tuned version of Bert, while in [3] it was reported a very poor result. Table 5 summarizes the main results we found.

Model	**Accuracy**	**Precision**	**Recall**	**F1**
BigBird (4096)	0.81	0.80	0.76	0.77
Bert (512)	0.87	**0.90**	0.81	0.84
HierBert (4096)	**0.88**	0.86	**0.81**	0.83
HAN [3]	–	0.88	0.78	0.81
Bert [3]	–	0.24	0.50	0.17
HierBert [3]	–	0.90	0.79	0.82

Table 5: Results of our models, compared with results in [3]. In [3], accuracy of the models is not reported.

In Figure 5 we also show how the accuracy HierBert is related with its maximum number of chunks. The accuracy is consistent in the different versions, but we notice a slight drop in performance for 4 chunks. The results are overall the best for the 8 chunk HierBert. It's interesting to note that even using only 512 tokens (1 chunk) gives us good results, this finding is highly in contrast to those in the original paper [3], so we think that there have must been errors in the reporting of the results.

From Figure 6 and 7 it's clear a relationship between the length of the document (chunk size) and the predictions. The shorter documents have more correct predictions than the longer ones. In 6 we notice how most of the documents have 8 chunks, that is caused by the model having a maximum number of chunk allowed in input. Basically all the documents longer than 8 chunk is truncated to its first 8 chunk.

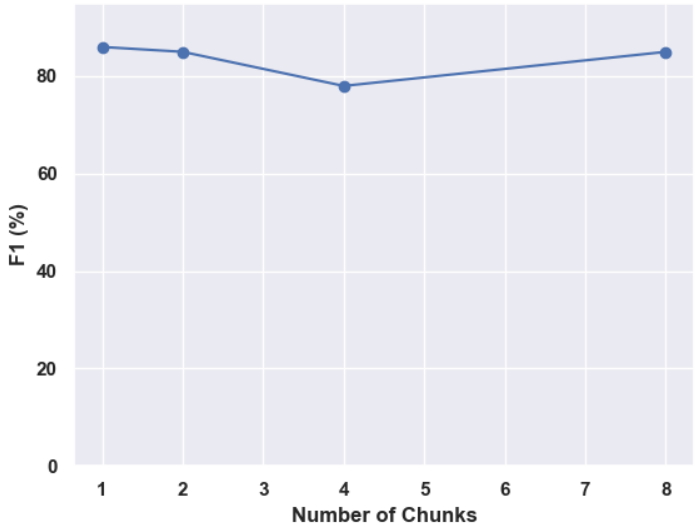


Figure 5: HierBert accuracy with different maximum input chunk size.

Figure 7 helps us to visualize better the negative trend. It would be worth it investigating the performances of HierBert with higher chunk limits, but this requires higher powered hardware due to memory issue.

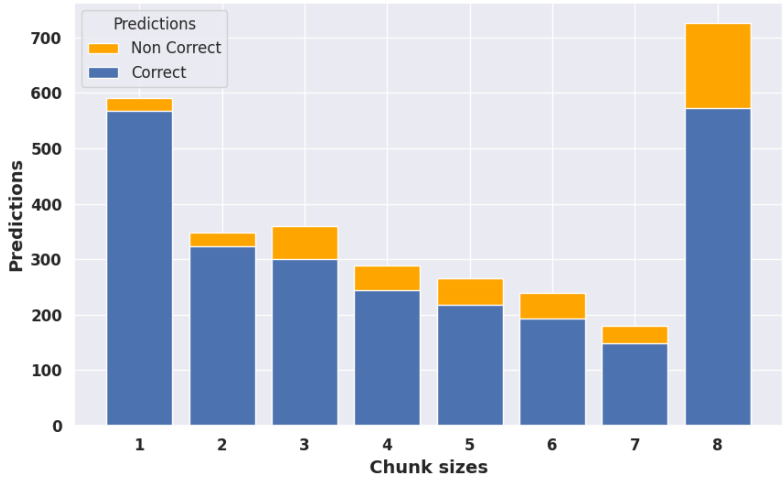


Figure 6: Predictions for each chunk size from the HierBert model with 8 chunk.

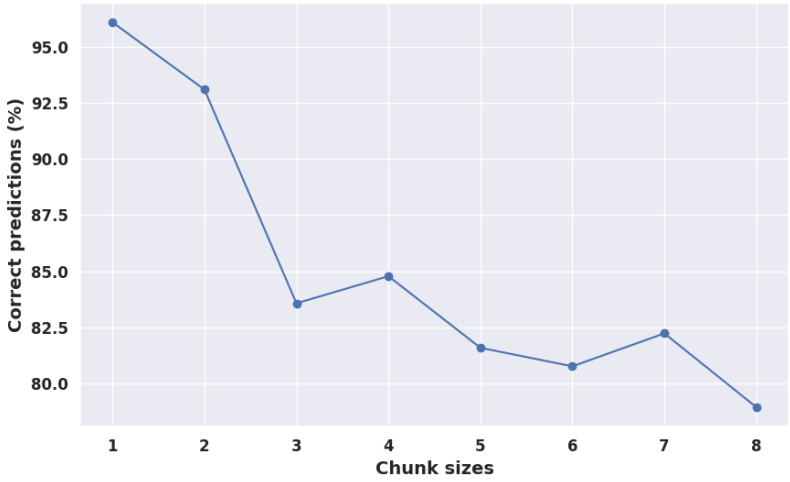


Figure 7: Percentage of correct predictions for each chunk size from the HierBert model with 8 chunk.

8 Conclusions

In this work, we have explored a new task of NLP in the legal domain, focusing particularly on the classification of legal documents. The biggest challenge was the considerable length of these documents, which often surpasses the token limits of standard NLP models. To overcome this obstacle, we adopted a hierarchical variant of BERT proposed by Chalkidis et al. [3]. This involved breaking down documents into smaller, more manageable parts and using a hierarchical attention mechanism to understand how these parts relate to each other.

Through experiments, we assessed the method's effectiveness on the ECHR dataset. We tested various classification models, including traditional ones like Random Forest and SVM, as well as more advanced techniques such as LSTM and attention networks. Additionally, we explored sparse transformers, which are inspired by recent advancements in how computers understand text.

We found our implementation of the Hierarchical Transformer being the best in dealing with lengthy documents and achieving results comparable to other state-of-the-art models like those reported in the original paper. From the analysis emerged a relationship between the document length and the prediction accuracy. Without the hardware limitations, we suggest that further analysis could be conducted on bigger chunk sizes.

In conclusion, our report underlines the importance of considering document length in text classification tasks and explores different approaches. We hope that our findings will inform future studies and inspire further exploration of innovative approaches to address the challenges of this task.

9 Acknowledgements

We would like to express our sincere gratitude to Prof. Giuseppe Attardi for providing the necessary hardware that was fundamental to conduct the analysis of this report. Thanks to this project, we had the opportunity to explore the scope of LLMs, and the experiments we conducted not only increased our knowledge but also allowed us to become more proficient in the tools essential in this field.

Code

The code written for the project can be found online at the Github repository `https://github.com/lilf4p/hlt-project`.

References

[1] N. Aletras and I. Chalkidis. "Echr dataset." (2019), [Online]. Available: https://archive.org/details/ECHR-ACL2019 (visited on 04/22/2024).

[2] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. arXiv: 1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805.

[3] I. Chalkidis, I. Androutsopoulos, and N. Aletras, "Neural legal judgment prediction in english," *CoRR*, vol. abs/1906.02059, 2019. arXiv: 1906.02059. [Online]. Available: http://arxiv.org/abs/1906.02059.

[4] M. Medvedeva, M. Wieling, and M. Vols, "Rethinking the field of automatic prediction of court decisions," *Artificial Intelligence and Law*, vol. 31, no. 1, pp. 195–212, 2023. DOI: 10.1007/s10506-021-09306-3.

[5] I. Chalkidis, A. Jana, D. Hartung, *et al.*, *Lexglue: A benchmark dataset for legal language understanding in english*, 2022. arXiv: 2110.00976 [cs.CL].

[6] N. Aletras, D. Tsarapatsanis, D. Preotiuc-Pietro, and V. Lampos, "Predicting judicial decisions of the european court of human rights: A natural language processing perspective," English, *PeerJ Computer Science*, Oct. 2016, ISSN: 2376-5992. DOI: 10.7717/peerj-cs.93.

[7] H. Ugginface. "Huggingface." (), [Online]. Available: https://huggingface.co/models (visited on 04/22/2024).

[8] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "LEGAL-BERT: The muppets straight out of law school," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, T. Cohn, Y. He, and Y. Liu, Eds., Online: Association for Computational Linguistics, Nov. 2020, pp. 2898–2904. DOI: 10.18653/v1/2020.findings-emnlp.261. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.261.

[9] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. 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*, K. Knight, A. Nenkova, and O. Rambow, Eds., San Diego, California: Association for Computational Linguistics, Jun. 2016, pp. 1480–1489. DOI: 10.18653/v1/N16-1174. [Online]. Available: https://aclanthology.org/N16-1174.

[10] I. Beltagy, M. E. Peters, and A. Cohan, *Longformer: The long-document transformer*, 2020. arXiv: 2004.05150 [cs.CL].

[11] M. Zaheer, G. Guruganesh, A. Dubey, *et al.*, *Big bird: Transformers for longer sequences*, 2021. arXiv: 2007.14062 [cs.LG].

[12] I. Chalkidis, X. Dai, M. Fergadiotis, P. Malakasiotis, and D. Elliott, *An exploration of hierarchical attention transformers for efficient long document classification*, 2022. arXiv: 2210.05529 [cs.CL].

[13] S. Gugger, L. Debut, T. Wolf, *et al.*, *Accelerate: Training and inference at scale made simple, efficient and adaptable.* https://github.com/huggingface/accelerate, Accessed: 22-04-2022, 2022.