No Argument Left Behind: Overlapping Chunks for Faster Processing of Arbitrarily Long Legal Texts

Israel Fama[1*], Bárbara Bueno[1*], Alexandre Alcoforado[1], Thomas Palmeira Ferraz[2], Arnold Moya[1], Anna Helena Reali Costa[1]

[1]Escola Politécnica, Universidade de São Paulo (USP), São Paulo, SP, Brazil

[2]Télécom Paris, Institut Polytechnique de Paris, Palaiseau, France

{israelfama, barbarabueno, anna.reali}@usp.br

Abstract. *In a context where the Brazilian judiciary system, the largest in the world, faces a crisis due to the slow processing of millions of cases, it becomes imperative to develop efficient methods for analyzing legal texts. We introduce* `uBERT`*, a hybrid model that combines Transformer and Recurrent Neural Network architectures to effectively handle long legal texts. Our approach processes the full text regardless of its length while maintaining reasonable computational overhead. Our experiments demonstrate that* `uBERT` *achieves superior performance compared to BERT+LSTM when overlapping input is used and is significantly faster than ULMFiT for processing long legal documents.*

1. Introduction

Legal NLP can be defined as the application of Natural Language Processing (NLP) techniques within the legal domain. This subfield of NLP has been experiencing rapidly growing interest from both academia and industry: [Katz et al. 2023] reports a significant increase in the volume of publications, rising from fewer than 30 papers in 2013 to nearly 120 in 2022. Brazil possesses the largest judiciary system in the world, comprising 18,000 judges distributed across 91 courts. At the time of writing, there are more than 84 million ongoing legal cases [CNJ 2024]. These numbers indicate both the need and the opportunity for innovative solutions to manage and analyze vast amounts of legal data.

We turn our focus to Legal Judgment Prediction (LJP), which involves predicting court decisions. Although predicting decisions may be a complex task, we argue it can be reduced to a Text Classification task, which has seen a marked increase in studies [Li et al. 2022], fueled by advancements in deep learning. In particular, the Transformer architecture emerged as a paradigm shift [Hasan 2022] for many NLP tasks. However, it still has limitations when handling long texts, which poses significant challenges in the legal domain, where documents are usually long and complex.

There is fruitful research being done on enhancing the input size limitation for Transformers, such as Retrieval-Augmented Language Models (RALMs) [Guu et al. 2020]. Current retrieval techniques, however, often trust embedding models which also can be sub-optimal when dealing with legal documents, where a single word in the whole document can make a difference. Also, these methods demand substantial computational resources and large document stores to achieve good performance.

[*]These authors contributed equally to this work.

Other methods combine input in a sequential way, often leveraging properties of Recurrent Neural Networks to process longer sequences [Wan et al. 2019], although those will also usually truncate the text if it is too long. But for documents in the legal domain, such as judicial decisions, most of the documents are usually composed of reasoning from the judge. Therefore, it is of our interest to have a method that uses the full text as input.

In this paper, we propose `uBERT`, a hybrid model that combines an encoder-based Transformer with a Recurrent Neural Network, capable of processing long texts. We propose an experimental setup with data from legal decisions, and compare `uBERT` to baselines BERT+LSTM, Big Bird and ULMFiT in the classification task. Our results show that `uBERT` slightly outperforms BERT+LSTM as long as overlapping input is introduced. Also, ULMFiT performs better for long texts, but is 4x slower than `uBERT`.

The remainder of this paper is structured as follows: Sect. 2 reviews related work on Legal NLP and long text classification; Sect. 3 outlines our proposal, including the formalization of the target task and the introduction of our model; Sect. 4 outlines the experiments we setup to assess our model in terms of performance and efficiency. Finally, we present the results and conclude with a discussion of the findings.

2. Related Work

Transformer-Based Approaches for Long Text Processing in Legal NLP: Longformer [Beltagy et al. 2020] employs a sparse attention mechanism, extending the input size limit to 4096 tokens, which is eight times the limit of BERT [Devlin et al. 2019]). [Hoang et al. 2023] applied this architecture to classify legal texts from the Indian Legal Documents Corpus – ILDC [Malik et al. 2021], but they did not process the entire text.

[Pappagari et al. 2019] introduced RoBERT, a method that splits long texts into overlapping chunks for recurrent encoding. While similar in concept to our architecture, a direct comparison is not possible due to limited details on their overlap and recurrence strategies. Moreover, RoBERT was evaluated on shorter texts compared to our dataset.

The overlapping algorithm in our approach, `uBERT`, is a specific case of SlidingBERT's method [Zhang et al. 2023], with the stride set to half the overlap. Unlike SlidingBERT, where tokens can appear in multiple chunks, we limit overlaps to two chunks to reduce computational overhead while preserving context continuity. This choice is driven by efficiency, not language differences.

[Menezes-Neto and Clementino 2022] introduced BrCAD-5[1], a dataset designed for Legal Judgment Prediction (LJP), and evaluated three architectures for this task: ULMFiT [Howard and Ruder 2018], BigBird [Zaheer et al. 2020], and BERT+LSTM. ULMFiT, a transfer learning model that fine-tunes a pre-trained language model for downstream NLP tasks, was the only architecture capable of processing the entire text as input. BigBird, a sparse-attention model, addresses the 512-token limit by focusing on subsets of tokens, thereby reducing computational complexity, and was configured to handle texts up to 7,680 tokens. For BERT+LSTM, documents were split into 512-token chunks, with truncation applied to middle chunks if a document required more than 15. While simi-

[1] This dataset consists of decisions issued by the Brazilian Federal Small Claims Court (FSCC). These decisions can be appealed to the Appellate Panel (AP), which re-examines the case and either reverses or affirms the initial ruling. Each data point in BrCAD-5 represents the text of a decision issued by the FSCC. The task proposed by the authors is to predict whether the AP will reverse or affirm the initial ruling based on the decision text.

lar in approach, uBERT differs from BERT+LSTM in that it uses a chunk overlapping strategy and imposes no limit on the number of chunks, ensuring the entire text is utilized without truncation.

Critiques and Limitations in Legal NLP Research: The legal industry has been slow to adopt NLP advancements, relying heavily on manual work by lawyers. [Mahari et al. 2023] identify a key issue: Legal NLP research often fails to align with the practical needs of legal practitioners. [Medvedeva and Mcbride 2023] further highlight a significant gap in Legal Judgment Prediction (LJP) research, criticizing the use of poorly designed datasets that rely on biased case facts extracted from judgments. This approach leads to models with overly optimistic performance that offer limited practical value to legal practitioners.

This work aims to bridge the gap between research and practice in the field of Legal NLP. We propose an architecture capable of processing virtually infinite-length legal texts and evaluate it on the BrCAD-5 dataset, which [Medvedeva and Mcbride 2023] regard as a well-designed benchmark.

3. Proposal

Text classification can be formalized as follows. Given a document d that represents a judicial decision, the goal is to make a prediction $y \in \{0, 1\}$, by learning a binary classifier f such that $f(d) = y$. The positive class $y = 1$ represents a decision that will be reversed by an Appellate Panel (AP). Since legal documents are often long, when using Transformer-based models, conventional approaches usually truncate text from d, which is sub-optimal for the task [Pappagari et al. 2019]. This can hinder performance on the Legal Judgment Prediction task, since some relevant part of the text may be cut off.

Believing that the text as a whole is more useful when learning a classifier, we propose unlimited BERT, or uBERT, an efficient architecture that combines an encoder-based Transformer with a Recurrent Neural Network, utilizing an overlapping algorithm during both training and inference to handle an unlimited number of input tokens. This approach is similar to the BERT+LSTM model used by [Menezes-Neto and Clementino 2022], but introduces modifications to maintain local context (through overlapping chunks) and accommodate documents of virtually any size. Although the quadratic memory complexity of the self-attention mechanism presents a challenge for scaling input indefinitely, we leverage the RNN's capacity to process long sequences, enabling it to take chunk embeddings and output a comprehensive document embedding. Several studies, such as [Hoang et al. 2023], have explored the combination of attention mechanisms and recurrence. Our model builds on this concept but applies overlapping during both training and inference, and does not limit the number of chunks processed by the encoder.

Figure 1 depicts the uBERT architecture. It shows key aspects to understand how our model works.

Let E be an encoder-based Transformer, with dim being the dimensionality of the output vector of the final layer, and R be a Recurrent Neural Network. Let max_{tok} be the maximum number of tokens E can process as input. Let max_c be the maximum number of chunks of max_{tok} tokens that E can process in parallel with a single run. We split document d into n chunks of size max_{tok} tokens, starting in the first token. For each

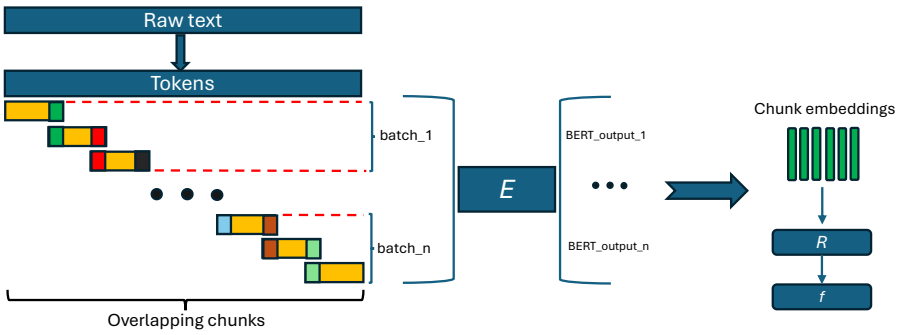


Figure 1: `uBERT` architecture.

run, we extract the hidden states from the last four layers of E, and concatenate them to form the representation of each chunk. This is based on the idea that different layers capture different linguistic features [Tenney et al. 2019]. Specifically, BERT+LSTM, the baseline most similar to our proposed architecture, extracts the hidden states from the last four layers. While other layers could be used for extraction, we retained this approach for consistency in model comparison. In each single run, we process $[1, max_c]$ chunks in parallel, generating $[1, max_c]$ vectors of embeddings, each with dimensionality $4 \times dim$. We iteratively process chunks from d until an embedding vector is generated for each chunk and thus preserving the entire text content of d.

Then, we concatenate the embedding vectors maintaining the order of the respective chunks, generating a tensor of dimensionality $(n, 4 \times dim)$. We process this tensor with the RNN sequentially, capturing the dependencies between them and generating a contextually enhanced representation for each chunk.

Splitting text by token count can disrupt its flow, so we use token overlap between chunks during both training and inference to maintain continuity. This technique, similar to that used by [Hoang et al. 2023] but applied more broadly, helps preserve the text's natural structure.

Our token overlap algorithm can be formalized as follows. Consider the judicial decision d as the tokenized sequence $S = \{t_1, \ldots, t_k\}$, where k is the number of tokens in d. We define the overlap size, z, as the number of tokens each chunk shares with its adjacent neighbors. Thus, any chunk shares $\lfloor \frac{z}{2} \rfloor$ tokens with the previous chunk and $\lfloor \frac{z}{2} \rfloor$ with the subsequent one. The first and last chunks, having only one adjacent chunk, share $\lfloor \frac{z}{2} \rfloor$ tokens with their respective neighbors.

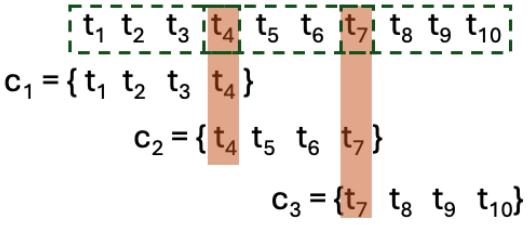


Figure 2: Overlapping chunks example.

Figure 2 provides a simple example for clarification. In this example, the chunk

size is 4, and $z = 2$. As shown, chunk $c_2 = \{t_4, t_5, t_6, t_7\}$ shares token t_4 with chunk c_1 and token t_7 with chunk c_3. Note that the first and last chunks share only one token with the neighboring chunk.

4. Experiments

In this section, we design experiments to assess our proposed model, `uBERT`, and validate its effectiveness on the legal domain. We split our experiments into 3, one for each of the following research questions:

RQ1: Would an encoder-based model benefit from using the entire text in terms of performance improvement?

We examine the impact of processing the whole documents using multiple encoder passes. We first tested if simply increasing text chunks to process the whole text without using overlap (`uBERT_0`) improves performance over BERT+LSTM, which processes only partial text in a single pass. Then, we investigated the effect of introducing overlaps (0 to 510 tokens[2]) between chunks to observe if the added local context enhances predictions.

RQ2: If performance improves, does it come with reasonable computational overhead?

We compare the inference time of our architecture against all baseline models to determine if it offers a performance gain and to assess the associated computational overhead.

RQ3: Is our architecture better for processing longer texts?

We explore the relationship between document length and model performance. We tested the models on the full test set as well as on its subsets, the 10% and the 1% longest texts. This experiment involved statistical analysis to determine whether longer texts lead to better or worse predictions.

Data: We used the BrCAD-5 dataset[3] in our experiments. The task is a binary classification, with Class 1 indicating that the AP reverses the previous decision, and Class 0 indicating it affirms. The dataset is imbalanced, with 22% of the data points belonging to Class 1. Although this imbalance ratio is consistent across all dataset splits, it varies significantly with text length.

Models: In this work, our model (`uBERT`) uses BERT as the encoder and LSTM as the RNN, with max_{tok} set to 512 tokens and up to 15 chunks (max_c) processed in parallel. Our training procedure follows the approach of [Menezes-Neto and Clementino 2022], where we fine-tune the last layer of BERT and the LSTM. The fine-tuning is conducted for 1 epoch utilizing the One Cycle learning rate scheduler. Our inference procedure mirrors the training process.

Baselines: our baseline models are ULMFiT (forward, backward and bidirectional)[4], Big Bird and BERT+LSTM. Notably, only ULMFiT and `uBERT` process the full text.

[2]The typical input size for BERT models is 512 tokens. Our overlap algorithm first slices the text and distributes the tokens. Only after this process are the special tokens [CLS] and [SEP] added, resulting in the well-known 512-token limit.

[3]This dataset is divided in training, validation, and test sets: the training set includes 380,673 documents, while the validation and test sets contain 76,342 and 76,299 entries, respectively.

[4]ULMFiT incorporates a forward language model (predicting the next token), a backward language model (predicting the previous token), and a bidirectional model that combines the two, allowing it to capture contextual information in both directions.

Computational Infrastructure and Resources: the experiments were conducted using Google's Colab infrastructure, specifically an NVIDIA A100 GPU with 40 GB of RAM.

Evaluation Metrics: We evaluate all models using the Macro F1 score and Matthews Correlation Coefficient (MCC). The Macro F1 score is a well-established metric across NLP fields, representing the harmonic mean of precision and recall, while MCC, though less common, is frequently used in the Legal Judgment Prediction (LJP) subfield, as noted by [Cui et al. 2022]. MCC measures the correlation between predicted and actual classifications by accounting for true positives, true negatives, false positives, and false negatives, making it suitable for imbalanced classes[5]. Additionally, MCC is the metric used by [Menezes-Neto and Clementino 2022], making it necessary for us to use it as well for model comparison. To compare different baselines and configurations of our `uBERT` model, we employed bootstrap resampling to obtain 95% confidence intervals, followed by Wilcoxon-Holm post-hoc analysis to assess statistical significance with $\alpha = 5\%$, following similar approaches [Demšar 2006, Zhu et al. 2020, Ferraz et al. 2021].

5. Results

Table 1 presents the results for all model configurations on the full test dataset, as well as the 10% and 1% longest texts. The baseline models were not run on the full test set in this study due to computational resource limitations. The results reported here are reproduced from [Menezes-Neto and Clementino 2022], which is why Table 1 does not include inference times for the full test set. Figure 3 displays the macro F1-scores across varying text lengths, while Figure 4 ranks the models using the MCC metric. Although MCC is effective for within-dataset comparisons, it is less suitable across datasets with differing class imbalance; hence, we rely on the macro F1-score for cross-dataset comparisons.

Table 1: uBERT performance across various overlap sizes compared with baselines.

	Full Test Set *(76.299 documents)*		**10% Set** *(7.632 documents)*			**1% Set** *(763 documents)*		
Dataset:	*Imbalance Ratio = 0.28*		*Imbalance Ratio = 0.32*			*Imbalance Ratio = 0.54*		
Architecture	**Macro-F1↑**	**MCC↑**	**Macro-F1↑**	**MCC↑**	**Inf.Time↓**	**Macro-F1↑**	**MCC↑**	**Inf.Time↓**
Baselines								
ULMFiT - fwd	65.1 %	0.32	64.9 %	0.32	1h 18min	72.3 %	0.47	11min:22s
ULMFiT - bwd	65.7 %	0.35	63.4 %	0.35	1h 18min	59.9 %	0.33	14min:44s
ULMFiT - bidir	66.9 %	0.37	64.8 %	0.34	1h 18min	69.3 %	0.43	14min:44s
Big Bird	52.0 %	0.27	44.0 %	0.23	22min	30.0 %	0.08	2min:58s
BERT+LSTM	64.1 %	0.33	63.2 %	0.31	12min	64.0 %	0.36	1min:29s
Ours								
uBERT_0	63.9 %	0.33	62.6 %	0.31	13min	61.3 %	0.34	1min:51s
uBERT_150	63.3 %	0.32	62.2 %	0.30	14min	59.4 %	0.32	2min:04s
uBERT_205	64.7 %	0.35	62.6 %	0.31	15min	62.0 %	0.35	2min:09s
uBERT_300	64.7 %	0.35	63.0 %	0.31	17min	63.0 %	0.36	2min:23s
uBERT_408	64.0 %	0.33	63.2 %	0.32	19min	64.3 %	0.38	2min:42s
uBERT_510	64.6 %	0.35	63.0 %	0.31	21min	64.2 %	0.38	3min:08s

[5]MCC ranges from -1 to 1, where 1 indicates perfect prediction, 0 indicates no better than random chance, and -1 indicates total disagreement between prediction and observation.

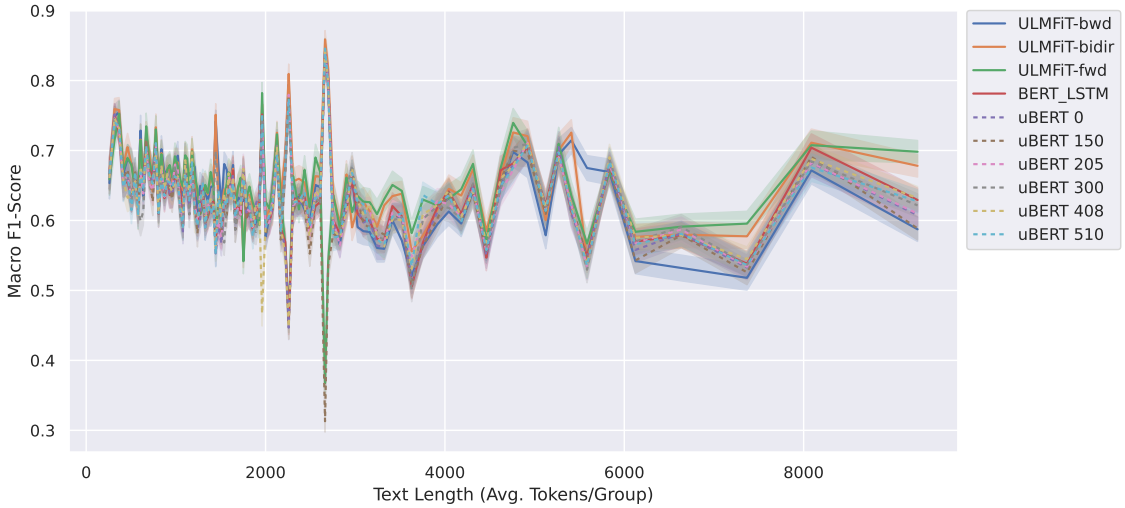
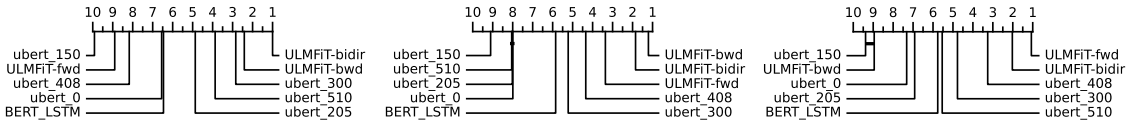


Figure 3: Macro-F1 score x Avg. Tokens/Group across different groups of same size ranked by the length.The error bars represent 95% confidence intervals obtained with bootstrap resampling.



(a) All Text **(b) 10% Longest Text** **(c) 1% Longest Text**

Figure 4: Critical difference diagram showing pairwise statistical comparison between baselines and varying overlap sizes for uBERT using the MCC. Connecting bars represent no statistical difference between methods.

Processing the Full Text Requires Overlap Comparing the BERT+LSTM baseline, which middle-truncates text when it exceeds input size, with our uBERT without overlap (uBERT_0), which uses the full text, we found that uBERT either underperformed or matched the baseline across all metrics. Notably, it performed worse on the 1% longest texts, where middle-truncation by BERT+LSTM occurs. **This suggests that merely processing the entire text is insufficient for longer inputs**. We hypothesize that non-overlapping chunks introduce noise due to abrupt segmentation, which degrades performance. Our results support this, showing that introducing overlap in uBERT configurations improves both Macro-F1 and MCC scores. The following uBERT configurations outperformed BERT+LSTM with statistical significance: uBERT_300, uBERT_510 and uBERT_205 on full test set; uBERT_408 and uBERT_300 on 10% longest; and uBERT_408, uBERT_300 and uBERT_510 on 1% longest. Thus, **incorporating overlap is crucial for maintaining semantic consistency and improving performance on longer texts**.

uBERT with Overlap is still Significantly Faster than ULMFiT As expected, introducing overlap in the uBERT architecture increased computational time overhead. However, across the full dataset and the 10% longest texts, uBERT configurations delivered better results than the BERT+LSTM baseline with comparable inference times. Notably, uBERT_408 achieved a 4x faster inference than ULMFiT on the 10% longest texts. For the 1% longest texts, the increased length required two passes of uBERT_408[6], resulting

[6]With zero overlap, uBERT can process a maximum of 7,650 tokens in a single encoder pass. This limit arises because uBERT handles up to 15 chunks of 510 tokens each (excluding special tokens [CLS] and [SEP]). Therefore, documents longer than 7,650 tokens require at least two encoder passes.

in 1.8x slower inference compared to BERT+LSTM, which needed to middle-truncate in all cases. Despite this, `uBERT_408` slightly outperformed BERT+LSTM, narrowing the performance gap with ULMFiT while maintaining a faster inference, highlighting the efficiency and effectiveness of our approach. In summary, in all subsets, **uBERT configurations reduced the BERT+LSTM gap being significantly faster than ULMFiT**.

ULMFiT Outperforms uBERT on Longer Texts As shown in Figure 3, model differences become more clear with increasing text length. Big Bird consistently underperforms on longer texts, which is why it was excluded from the comparison charts. **While some uBERT configurations outperform BERT+LSTM on longer texts, F1 scores in both models degrade compared to full test dataset performance**. In contrast, ULMFiT models improve on longer texts compared to the full dataset. This suggests that **our architecture mitigates the degradation for longer text that is inherent to the BERT+LSTM approach, but still falls short of ULMFiT models, that handle better longer text but at a cost of 4x slower inference time**.

6. Conclusion and Future Work

Our experiments demonstrate that the `uBERT` model improves the handling of legal texts compared to baseline encoder-based models, particularly on longer texts, due to its capability to process entire documents using overlapping chunks. Despite the increased computational overhead, `uBERT` remains faster than ULMFiT. `uBERT` slightly outperforms BERT+LSTM, but still falls short of ULMFiT. Thus, further refinement is needed to fully match ULMFiT‘s performance. Notably, even ULMFiT, the top-performing model in our experiments, achieves relatively low Macro-F1 scores, suggesting that processing the full text alone is insufficient for high performance on this task. In this direction, future research should expand the evaluation methodology by analyzing correctly and incorrectly classified cases across all tested models to assess whether specific characteristics of judicial decisions make them more prone to misclassification by certain models. Such an analysis, however, requires a multidisciplinary approach, including expert input from highly skilled legal practitioners.

Future research should also explore different chunking strategies to enhance text processing. Comparing syntactic chunking, which is based on grammatical structure, with semantic chunking, which is based on content meaning, could provide valuable insights. As this study focuses on a Portuguese-language dataset, evaluating these chunking approaches across datasets in multiple languages would help determine if optimal chunking strategies vary with language, contributing to more robust long-text segmentation and model performance across diverse linguistic contexts.

Acknowledgements

This work was supported by *CAPES* (Finance Code 001), *CNPQ* (grant 312360/23-1), *Programa Unificado de Bolsas de Estudo para Apoio à Formação de Estudantes (PUB-USP)*, *USP-IBM-FAPESP Center for Artificial Intelligence* (FAPESP grant 2019/07665-4), and Secretaria da Fazenda do Estado do Rio Grande do Sul (SEFAZ-RS), Brazil.

References

Beltagy, I., Peters, M. E., and Cohan, A. (2020). Longformer: The long-document transformer.

CNJ (2024). Conselho nacional de justiça - cnj. Accessed: 2024-08-05.

Cui, J., Shen, X., Nie, F., Wang, Z., Wang, J., and Chen, Y. (2022). A survey on legal judgment prediction: Datasets, metrics, models and challenges. *arXiv preprint arXiv:2204.04859v1*.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805v2*.

Ferraz, T. P., Alcoforado, A., Bustos, E., Oliveira, A. S., Gerber, R., Müller, N., d'Almeida, A. C., Veloso, B. M., and Costa, A. H. R. (2021). Debacer: a method for slicing moderated debates. In *Anais do XVIII Encontro Nacional de Inteligência Artificial e Computacional*, pages 667–678. Sociedade Brasileira de Computação-SBC.

Guu, K., Lee, K., Tung, Z., Pasupat, P., and Chang, M.-W. (2020). Realm: Retrieval-augmented language model pre-training.

Hasan, M. (2022). Transformers in natural language processing.

Hoang, T. D., Bui, C. M., and Bui, N. (2023). Viettel-AI at SemEval-2023 task 6: Legal document understanding with longformer for court judgment prediction with explanation. In Ojha, A. K., Doğruöz, A. S., Da San Martino, G., Tayyar Madabushi, H., Kumar, R., and Sartori, E., editors, *Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023)*, pages 862–868, Toronto, Canada. Association for Computational Linguistics.

Howard, J. and Ruder, S. (2018). Fine-tuned language models for text classification. *CoRR*, abs/1801.06146.

Katz, D., Hartung, D., Gerlach, L., Jana, A., and Bommarito, M. (2023). Natural language processing in the legal domain. *arXiv preprint arXiv:2302.12039v1*.

Li, Q., Peng, H., Li, J., Xia, C., Yang, R., Sun, L., Yu, P. S., and He, L. (2022). A survey on text classification: From traditional to deep learning. *ACM Trans. Intell. Syst. Technol.*, 13(2).

Mahari, R., Stammbach, D., Ash, E., and Pentland, A. (2023). The law and NLP: Bridging disciplinary disconnects. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3445–3454, Singapore. Association for Computational Linguistics.

Malik, V., Sanjay, R., Nigam, S. K., Ghosh, K., Guha, S. K., Bhattacharya, A., and Modi, A. (2021). ILDC for CJPE: Indian legal documents corpus for court judgment prediction and explanation. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4046–4062, Online. Association for Computational Linguistics.

Medvedeva, M. and Mcbride, P. (2023). Legal judgment prediction: If you are going to do it, do it right. In Preoțiuc-Pietro, D., Goanta, C., Chalkidis, I., Barrett, L., Spanakis, G., and Aletras, N., editors, *Proceedings of the Natural Legal Language Processing Workshop 2023*, pages 73–84, Singapore. Association for Computational Linguistics.

Menezes-Neto, E. J. d. and Clementino, M. B. M. (2022). Using deep learning to predict outcomes of legal appeals better than human experts: A study with data from brazilian federal courts. *PLOS ONE*, 17(7):1–20.

Pappagari, R., Żelasko, P., Villalba, J., Carmiel, Y., and Dehak, N. (2019). Hierarchical transformers for long document classification.

Tenney, I., Das, D., and Pavlick, E. (2019). Bert rediscovers the classical nlp pipeline.

Wan, L., Seddon, M., Papageorgiou, G., and Bernardoni, M. (2019). Long-length legal document classification. *arXiv preprint arXiv:1912.06905v1*.

Zaheer, M., Guruganesh, G., Dubey, A., Ainslie, J., Alberti, C., Ontañón, S., Pham, P., Ravula, A., Wang, Q., Yang, L., and Ahmed, A. (2020). Big bird: Transformers for longer sequences. *CoRR*, abs/2007.14062.

Zhang, L., Wang, W., Yu, K., huang, J., Lyu, Q., Xue, H., and Hetang, C. (2023). Sliding-bert: Striding towards conversational machine comprehension in long contex. *Adv. Artif. Intell. Mach. Learn.*, 3:1325–1339.

Zhu, H., Mak, D., Gioannini, J., and Xia, F. (2020). NLPStatTest: A toolkit for comparing NLP system performance. In Wong, D. and Kiela, D., editors, *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 40–46, Suzhou, China. Association for Computational Linguistics.