# Effect of Hierarchical Domain-specific Language Models and Attention in the Classification of Decisions for Legal Cases

Nishchal Prasad[1,*], Mohand Boughanem[1] and Taoufiq Dkaki[1]

[1]*Institut de Recherche en Informatique de Toulouse (IRIT), Toulouse, France*

**Abstract**

In order to automate a judicial process, a model to accurately predict the best probable decision of a legal case from the facts is desired. We try to explore this task of decision prediction on unannotated and unstructured large legal documents with only the results of the decision. For this task, we explored many available deep learning architectures including transformer-based language models (BERT, XLNet), domain-specific language model (LEGAL-BERT), attention mechanism, and sequence models (LSTM, GRU). With the different combinations of these architectures and methods, we ran extensive experiments upon an English legal dataset called ILDC and developed many hierarchical domain-specific language models all of which improves the performance by at least 2 metric points, with the best amongst them giving an improvement of approximately 3 metric points on the previous baseline models on this dataset, showing that the domain-specific models; when fine-tuned; adapts well to a domain of the same nature but with a different syntax, lexicon and grammar setting, and improves the performance significantly.

**Keywords**

LEGAL-BERT, Domain-specific Large Document Classification, Legal Case Prediction, Large Unstructured Documents

## 1. Introduction

A mechanism to assist judges and courts to reach a conclusion for the outcome of an ongoing legal case, is sought after for many years [1]. Also with the One of the major milestones to develop such a robust mechanism for practical legal assistance is the prediction of court judgments in a real-life setting, i.e. predicting the best probable decision from only the previous case arguments and case facts. This can help to propel the slow judicial process which plagues the judicial system of many countries. One such example can be seen in the Indian judicial system. [1] A solution to this problem of legal case decision prediction, can also help to cut the cost of case proceedings for people unfamiliar to intensive judicial system and law articles, by giving useful decision results and insights into their legal cases. This will give the courts the required time and space to develop other branches of the judicial process and also de-congest

*CIRCLE (Joint Conference of the Information Retrieval Communities in Europe), July 04–07, 2022, Samatan, Gers, France*

*Corresponding author.

Nishchal.Prasad@irit.fr (N. Prasad); Mohand.Boughanem@irit.fr (M. Boughanem); Taoufiq.Dkaki@irit.fr (T. Dkaki)

https://www.irit.fr/~Mohand.Boughanem/ (M. Boughanem)



[1]www.tribuneindia.com/news/archive/comment/backlog-of-cases-crippling-judiciary-776503

tribunal court cases.

Since the legal documents are mostly language-oriented in the form of complex legal texts, the task of decision prediction has been formulated as a text classification task. But as compared to a general text classification problem this task of legal decision prediction is more complex and sophisticated. This is due to many reasons involving the unstructured and unannotated noisy textual representation of legal case proceedings, which makes the process of automatically extracting the arguments and facts from the case proceedings difficult. The legal text also differs from a standard text in terms of lexical understanding having a very specific vocabulary and complex document structure, which requires adapting the pre-trained models (trained on general text) on legal texts.

In this paper, we try to confront the problem of decision prediction from legal texts by developing deep learning methods. We aim to predict the final decision of a legal case from its facts and arguments in unannotated and unstructured legal documents, which replicates the real-life setting of legal case documents. We work only in the development of a robust predictor while the work on the explanation of the predictions is underway at the time of writing this paper. Although this is not a novel task in itself, it is our first step to developing an architectural model for legal understanding, and decision prediction. We have explored the effect of a domain-specific language model (LEGAL-BERT [2]) over the general ones and have provided the experimental results for the same. While our work has dealt in the context of legal texts in the English language, the findings of this work can be leveraged to be adapted to legal texts in any language, on the condition that there is a sufficiently large clean dataset in the same language for the models and methods to be adapted (trained) on.

The main contributions of this paper is summarized as below:

- Legal judgment prediction model:
  We propose a baseline model for legal judgment prediction which hierarchically builds upon domain-specific BERT [3] known as LEGAL-BERT [2] and two-layered Bi-LSTM with multi-head scaled dot-product attention [4], which achieves significant higher metric scores on the previous baseline models. The model is based on the hypothesis that a domain specific pre-trained language model is transferable in same domain. This hypothesis is also supported by the experimental results in the following sections.
- Experimental approaches:
  We have explored the ILDC dataset [5] and experimented with state-of-the-art architectures involving recursive neural networks (GRU, LSTM, CNN), transformers (BERT, XLNet) and attention mechanisms on a dataset of large unstructured and unannotated legal documents.
- Evaluations:
  We performed extensive experiments on the ILDC dataset with different baseline models and improved upon their architectures to develop a final proposed baseline architecture; which achieves a significantly higher metric score upon the task for which the previous baseline architectures were trained on; showing that fine-tuning of pre-trained domain-specific language models helps to adapt and give a better understanding of a similar domain language having different lexicon, grammar and syntactical setting.

## 2. Related Work

Several research with the methods of machine learning and deep learning have been conducted in the past on the problem of automatic predicting the outcome of a legal case, alongside providing different approaches, methods and corpora suited to individual prediction tasks. In 2018 Xiao et al. [6] released the Chinese AI and Law challenge dataset (CAIL2018) for legal judgment prediction which contains rich annotations for the judgments of more than 2.6 million criminal cases. This dataset consists of detailed annotations to the related law articles to cases, the prison terms and the charges. Chalkidis et. al. [7] introduced a dataset from the case proceedings of European Cour of Human Rights, in English, where each case has a score which states its importance. They described a Legal Judgment Prediction (LJP) task for their dataset which aims to predict the outcome of a legal case with the annotated case facts, and law violations. For this task, they proposed a hierarchical version of BERT [3] to tackle BERT's limitation of a fixed number of input tokens. Zhong et al. [8] proposed TOP-JUDGE which formulates the dependency among the subtasks of legal judgment prediction through Directed Acyclic Graphs (DAG) by attending to the relation between different subtasks of the judgment prediction through topological multi-task learning. Luo et al. [9] defined a charge prediction task from the case facts of a Chinese criminal case dataset and proposed an attention-based method to predict the same along with relevant law articles. Zhong et al. [10] proposed QAjudge, based on reinforcement learning to predict the outcome of a legal case from the facts by visualizing the process giving interpretable judgments. Chen et al. [11] proposed a Deep Gating Network (DGN) to predict the prison term for criminals based on the criminal charges and the case facts.

While much of the research focuses on the legal case prediction for a specific setting (such as civil, criminal) with rich annotated cases providing good learning parameters for helping the decision classification, we focus more on predicting the outcome of the general legal cases from large unannotated and unstructured legal documents. Malik et al. [5] introduced a dataset named the Indian Legal Document Corpus (ILDC) and experimented upon it to provide a baseline model with their Case Judgment Prediction and Explanation (CJPE) task which achieves a macro-F1 score of $77.79\%$ and an accuracy of $78\%$ in the judgment prediction task.

CJPE is somewhat similar to our task while we aim to leverage our task to French legal documents in the future with more focus to cluster the case documents to their specific types. Because of the similarity of the ILDC dataset with the dataset requirement for our first task (of predicting decisions from unstructured legal documents), we develop, experiment, and evaluate our classification models on the ILDC dataset contributed by Malik et al. [5].

## 3. Methods

We formulate this task of legal judgement prediction as a text classification problem, given below:

For an unstructured legal case document 'C', predict its decision 'D' among the two labels 'accepted'$(= 1)$ and 'rejected'$(= 0)$, given only the facts of the legal document.

To move forward with the classification task we experimented with several deep learning architectures and methods detailed hereafter.

### 3.1. Sequence-to-sequence RNN encoders:

We experimented with some of the Recursive Neural Networks (RNN) such as GRU [12] and LSTM [13] with bidirectional nature [14] to process the sequence information in both forward and backward directions. Since the ILDC dataset consists of large documents of variable lengths, each having several sentences (tens of thousands of tokens in total), it becomes computationally complex and expensive to process and determine the embeddings of all individual words as a sequence of sequence i.e. words in sentences in a document for all documents. So instead we resort to encoding the sentences as sequences in a document (i.e sequence of encoded/vectorized sentences). To encode the sentences in the documents we have used separately two state-of-the-art pre-trained sentence encoders namely Universal Sentence Encoder [15] and S-BERT tokenizer [16], trained on general texts. We divided the documents into chunks (with the idea that these chunks can be treated as a near estimate for the sentences in the documents) with overlaps to count for the miss in the sentence breaks while dividing/chunking. These are passed into the encoders to obtain the chunk embeddings. These chunk embeddings for a document are concatenated together for further processing.

We used Bi[2]-LSTM[3] (or Bi[2]-GRU[4]) with two layers and dropouts in between with further feed-forward layers for classification.

### 3.2. Transformer Encoders:

Pre-trained transformer [4] encoders such as BERT [3], XLNet [17] have shown significant improvements in language modeling and understanding and can be adapted for downstream tasks with fine-tuning of the internal kernel weights or pre-training it altogether in a domain-specific task either from scratch or from a previous pretrained checkpoint. In our work we have experimented with BERT-base[5] and XLNet-base[6] trained on general text by fine-tuning on the training set. We have used max-pooling on the output of the final layer to get a document-level representation as an input to a feedforward network for classification. Since the text in the legal domain has a specific lexicon, vocabulary and differs in syntax from the general text, the sentence and document embeddings generated from the models pre-trained on general text may not properly adapt to the domain-specific context. Hence we also tried to check this argument with a BERT model pre-trained on legal text, known as LEGAL-BERT [2]. The same architecture of max-pooling and feedforward network was used to compare with the results from previous models of BERT and XLNet, which can be found in the Table 3. A document is divided into smaller chunks with overlap (as in Section 3.1), each having 512 tokens including the [CLS] and [SEP] token [3]. These chunks are then passed into the tokenizer (of the respective transformer encoder model), and from its output embeddings, the CLS tokens are extracted and taken as the vectorized representation for the chunks. These are then concatenated together to form the tokenized representation for the document to be used as the input for the transformer encoder model.

[2] www.tensorflow.org/api_docs/python/tf/keras/layers/Bidirectional
[3] www.tensorflow.org/api_docs/python/tf/keras/layers/LSTM
[4] www.tensorflow.org/api_docs/python/tf/keras/layers/GRU
[5] https://huggingface.co/docs/transformers/model_doc/bert
[6] https://huggingface.co/docs/transformers/model_doc/xlnet

### 3.3. Hierarchical Transformers (Transformer Encoder + RNN):

We used a hierarchical transformer method taking the idea from [18]. The document is divided into chunks (with or without overlaps) of a fixed length of 512 tokens including the [CLS] and [SEP] tokens. Each chunk is passed into the tokenizer to obtain the tokenized representation to be used as input to the respective transformer encoder model. The output from the last layer of the transformer encoder model is max-pooled to get the [CLS] representation for the chunk. Each of these [CLS] representations are accumulated together to form the new sequence to be used as the embedding for further processing with sequence encoder layers(Bi-GRU, BiLSTM, etc.) for classification. The details of the model architecture for the hierarchical transformer can be seen in the Table 1. LEGAL-BERT fine-tuned on ILDC$_{multi}$ is used to extract the [CLS] representations, owing to its better performance as compared to other transformer architectures (Table 3). Also, it can be argued that even though the LEGAL-BERT is pretrained on US/EU legal texts and not on the Indian legal texts (which differ in lexicons and syntax) [5], its fine-tuned model can be adapted to the respective setting in the same way as other pre-trained models trained on general texts are used (with fine-tuning) for domain-specific downstream tasks (as can be seen in the experimental results in Table 3). In general, we experimented with two different types of setup in this architecture:

- **Without attention:** The accumulated [CLS] vectors are taken as embedding inputs to the sequence models used in Section 3.1, which consists of the general setup of two layers of either Bi-GRU, Bi-LSTM or their combination. Dropouts we also introduced between the bidirectional layers to increase randomization and prevent overfitting.
- **With attention:** Dot-product attention [19] was used with the Bi-LSTM (layer 2) output as the query and key-value pair. Multi-head scaled dot-product attention [4] with the outputs of Bi-LSTM were also used. We used different combinations of query and key value pairs for the multi-head scaled dot-product attention which are:
  - The accumulated [CLS] representations for a document as the query, and the output from Bi-LSTM (layer 1) as the key-value pair.
  - The output from Bi-LSTM (Layer 1) as the query, and Bi-LSTM (layer 2) output as the key-value pair.
  - Bi-LSTM (layer 2) output as the query and key-value pair.

## 4. Experimental Setup and Hyperparameters

For all the experiments and architecture development we used the Tensorflow[7] framework, pandas [8] and NumPy [9] library. The pre-trained transformer models were taken from the HuggingFace[10] library. The experiments were run on Colab[11] with an Nvidia[12] Tesla P100(16GB)

[7] www.tensorflow.org
[8] https://pandas.pydata.org/
[9] https://numpy.org/
[10] https://huggingface.co/
[11] https://colab.research.google.com/
[12] https://www.nvidia.com/

**Table 1**
Architectural and hyper-parametric details of the models

| **Models** | | **Hyper-parameters and architectural details.**<br>$e$ = number of epochs,<br>$h$ = number of attention heads,<br>$e_D$ = embedding dimension,<br>$Q$ = query, $K$ = key, $V$ = Value, $n_L$ = RNN layers,<br>$L_o^i$ = output from $i^{th}$ RNN layer,<br>$L_a$ = activation for RNN layers,<br>$L_u^i$ = units in $i^{th}$ RNN layer,<br>$n_F$ = number of feed forward layers,<br>$F_d^i$ = dimension of $i^{th}$ feed forward layer,<br>$a_f^i$ = activation function for $i^{th}$ feed forward layer,<br>concat = concatenate,<br>drop(p) = Dropout (dropout percent) |
|---|---|---|
| **Sequence-to-sequence RNN encoders**<br>(train set = ILDC$_{single}$, ILDC$_{multi}$) | | $n_L = 2, L_a = tanh, L_u^1 = 100, L_u^2 = 100,$<br>$n_F = 2, F_d^1 = 30, F_d^2 = 1,$<br>$a_f^1 = ReLu, a_f^2 = sigmoid$ (for classification) |
| Universal Sentence Encoder + BiLSTM | | $e_D = 768, e = 3$ |
| Universal Sentence Encoder + BiGRU + Dropout(0.01) | | $e_D = 768, e = 6$ |
| S-BERT embeddings + BiLSTM | | $e_D = 384,\ e = 3$ |
| S-BERT embeddings + BiLSTM + Dropout(0.01) | | $e_D = 384,\ e = 6$ |
| **Pre-Trained Transformer Encoders** (train set = ILDC$_{multi}$) | | $n_F = 1, a_f^1 = sigmoid$ (for classification) |
| BERT | | + max-pooled BERT output + feed forward, $e = 2$ |
| XLNet | | + max-pooled XLNet output + feed forward, $e = 2$ |
| LEGAL-BERT | | + max-pooled LEGAL-BERT output + feed forward, $e = 2$ |
| **Hierarchical Transformers** (train set = ILDC$_{single}$, ILDC$_{multi}$): | | $n_L = 2, L_a = tanh, L_u^1 = 100, L_u^2 = 100,$<br>$n_F = 2, F_d^1 = 30, F_d^2 = 1,$<br>$a_f^1 = ReLu, a_f^2 = sigmoid$ (for classification) |
| LEGAL-BERT + ($fine\text{-}tuned$) | Bi-GRU | $e = 3$ |
| | Bi-GRU | $e = 10$ |
| | Bi-LSTM + Bi-GRU | $e = 6$ |
| | Bi-LSTM | $e = 6$ |
| | Bi-LSTM + Dropout(0.01) | $e = 6$<br>$e = 8$ (for ILDC$_{single}$) |
| | Bi-LSTM + Dropout + Dot-product attention | $Q, K, V = L_o^2, e = 6$ |
| | Bi-LSTM + Dropout + Multi-head attention$_\alpha$ (MHA) | $Q, K, V = L_o^2, h = 16, e = 6,$<br>concat(max-pool(MHA output,$L_o^2$)) to feed forward network |
| | Bi-LSTM + Dropout + Multi-head attention$_\beta$ | $Q = L_o^2$; $K, V$ = drop(0.01)$(L_o^2)$ , $h = 16, e = 6,$<br>concat(max-pool(MHA output,$L_o^2$<br>)) to feed forward network |
| | Bi-LSTM + Dropout + Multi-head attention$_\gamma$ | $Q = [CLS]$ representations; $K, V = L_o^2, h = 16,$<br>$e = 6,$<br>concat(max-pool(MHA output),drop(0.2)$(L_o^2)$) to feed forward network |

GPU. For all the experiments $sigmoid$ activation was used for classification in the last layer. 'Relu' activation function was chosen for the hidden feed-forward layers, while the sequence models use the $tanh$ activation function. Adam [20] was used as the optimization algorithm for training. As this is a problem of binary classification we use 'binary cross-entropy' as the loss function. To train the models we reduce the learning rate by a factor of $0.95$ based on the updates

**Table 2**
ILDC statistics describing the dataset split and imbalance

| | Split | Accepted : Rejected Cases | Label |
|---|---|---|---|
| ILDC$_{single}$ (7593 cases) | Train | 1935 : 3147 | 0 = Rejected<br>1 = Accepted |
| | Validation | 497 : 497 | |
| | Test | 762 : 755 | |
| ILDC$_{multi}$ (34816 cases) | Train | 13385 : 18920 | |
| | Validation | 497 : 497 | |
| | Test | 762 : 755 | |

on the monitored metric with patience in two epochs.[13] All the transformer models of Section 3.2 were fine-tuned for two epochs with a batch size of 10 documents. The hierarchical transformer architectures in Section 3.3, were trained on a batch size of 32 documents. Architectural specific details and other hyper-parameters can be found in Table 1.

## 5. Dataset description

We used the dataset[14] introduced by Malik et al. [5], which contains the case proceedings from the Supreme Court of India. For whether a claim(s) is 'accepted' or 'rejected' for a case(s) filed by the appellant in the Supreme Court of India is decided by a jury, which is taken as the label for the respective legal case document in the dataset. These labels are used to train the models/architectures in the experiments. The dataset has two parts ILDC$_{single}$ and ILDC$_{multi}$. ILDC$_{single}$ consists of those case proceedings for which there is a single decision for a petition or a same decision across all the multiple petitions. While the documents in ILDC$_{multi}$ are the more common case of case proceedings that involve multiple petitions with different decisions. The labeling of the documents in ILDC$_{multi}$ is taken as it is (stating the fact that computing multiple decisions for multiple petitions is computationally complex and expensive) where the label is set to be 'accepted' class if a single petition among the multiple appeals is 'accepted' otherwise it is set to the 'rejected' class. The dataset statistics are given in Table 2. We experiment with the same subsets of the dataset for training, validation, and testing as provided by the authors to maintain consistency in the experimental results and compare on the same test cases across all the experiments for decision classification.

## 6. Results and discussion

To measure the model performance we used the macro-precision, macro-recall, and macro-F1 scores as our performance metrics in order for the results to be comparable to the previous models on the same dataset. In Table 3 we omit the results of the pre-trained transformer models trained on ILDC$_{single}$, since we only use the pre-trained transformer models finetuned on ILDC$_{multi}$ for further development of the hierarchical models. Also since the number of

[13] https://keras.io/api/callbacks/reduce_lr_on_plateau/
[14] The dataset can be requested from its original authors [5]. We do not have the rights to circulate this dataset.

**Table 3**
Experimental results of legal case text classification task on different models

| Models | | metrics (metrics $\times 100$ *to get* $\%$ *values*) | | | |
|---|---|---|---|---|---|
| | | **Accuracy** | **Macro-F1** | **Macro-Recall** | **Macro-Precision** |
| **Sequence-to-sequence RNN encoders (train set = ILDC$_{single}$)** | | | | | |
| Universal Sentence Encoder + BiLSTM | | 0.5679 | 0.5731 | 0.5686 | 0.5778 |
| Universal Sentence Encoder + BiGRU + Dropout(0.01) | | 0.5712 | 0.5748 | 0.5718 | 0.5779 |
| S-BERT embeddings + BiLSTM | | 0.5528 | 0.5603 | 0.5536 | 0.5672 |
| S-BERT embeddings + BiLSTM + Dropout(0.01) | | 0.5528 | 0.5604 | 0.5537 | 0.5673 |
| **Sequence-to-sequence RNN encoders (train set = ILDC$_{multi}$)** | | | | | |
| Universal Sentence Encoder + BiLSTM | | 0.5574 | 0.5858 | 0.5779 | 0.5939 |
| Universal Sentence Encoder + BiGRU + Dropout(0.01) | | 0.5547 | 0.5606 | 0.5591 | 0.5622 |
| S-BERT embeddings + BiLSTM | | 0.56 | 0.5567 | 0.5558 | 0.5578 |
| S-BERT embeddings + BiLSTM + Dropout(0.01) | | 0.59 | 0.5893 | 0.5869 | 0.5918 |
| **Pre-Trained Transformer Encoders (train set = ILDC$_{multi}$)** | | | | | |
| BERT | | 0.6052 | 0.6322 | 0.6055 | 0.6613 |
| XLNet | | 0.7051 | 0.7103 | 0.7009 | 0.7201 |
| LEGAL-BERT | | 0.7383 | 0.7382 | 0.7384 | 0.7390 |
| **Hierarchical Transformers (train set = ILDC$_{single}$):** | | | | | |
| LEGAL-BERT + ($fine\text{-}tuned$) | Bi-GRU | 0.7961 | 0.8033 | 0.7966 | 0.8101 |
| | Bi-GRU | 0.8001 | 0.8043 | 0.8004 | 0.8082 |
| | Bi-LSTM + Bi-GRU | 0.7744 | 0.8029 | 0.8016 | 0.8041 |
| | Bi-LSTM | 0.80 | 0.8060 | 0.8018 | 0.8103 |
| | Bi-LSTM + Dropout(0.01) | 0.79<br>**0.81** | 0.7964<br>**0.8084** | 0.7881<br>**0.8063** | 0.8051<br>**0.8106** |
| | Bi-LSTM + Dropout + Dot-product attention | 0.79 | 0.7970 | 0.7893 | 0.8048 |
| | Bi-LSTM + Dropout + Multi-head attention$_{\alpha}$ | 0.80 | 0.8076 | 0.8031 | 0.8123 |
| | Bi-LSTM + Dropout + Multi-head attention$_{\beta}$ | **0.81** | **0.8125** | **0.8090** | **0.8160** |
| | Bi-LSTM + Dropout + Multi-head attention$_{\gamma}$ | 0.80 | 0.8069 | 0.8043 | 0.8095 |
| **Hierarchical Transformers (train set = ILDC$_{multi}$):** | | | | | |
| LEGAL-BERT + ($fine\text{-}tuned$) | Bi-GRU | 0.7915 | 0.7916 | 0.7916 | 0.7916 |
| | Bi-GRU | 0.7935 | 0.7943 | 0.7934 | 0.7953 |
| | Bi-LSTM + Bi-GRU | 0.8080 | 0.8015 | 0.7932 | 0.7981 |
| | Bi-LSTM | 0.80 | 0.8010 | 0.7999 | 0.8021 |
| | Bi-LSTM + Dropout(0.01) | **0.80** | **0.8035** | **0.8019** | **0.8052** |
| | Bi-LSTM + Dropout + Dot-product attention | 0.80 | 0.8007 | 0.7986 | 0.7997 |
| | Bi-LSTM + Dropout + Multi-head attention$_{\alpha}$ | 0.80 | 0.8002 | 0.7993 | 0.7998 |
| | Bi-LSTM + Dropout + Multi-head attention$_{\beta}$ | **0.81** | **0.8070** | **0.8066** | **0.8073** |
| | Bi-LSTM + Dropout + Multi-head attention$_{\gamma}$ | 0.80 | 0.7984 | 0.7967 | 0.7975 |

training instances is much less in ILDC$_{single}$ the fine-tuned transformer models yield less understanding as compared to ILDC$_{multi}$. As can be seen in Table 3, the sequence models with the pre-trained encoders (Universal Sentence Encoder and S-BERT encoder) have poor performance in all the performance metrics. This can be accounted for by the fact that these encoders are not fine-tuned during the model training process and also their embeddings are

more aligned to general texts rather than on the domain-specific to the legal texts. Even so, the embeddings from the Universal Sentence Encoder give slightly better performance than S-BERT embeddings both in ILDC$_{single}$ and ILDC$_{multi}$, without any architectural modifications (i.e. dropouts) to the baseline RNN layer. The pre-trained transformer models trained on general English texts improved the metric scores with BERT achieving a F1 score of $0.6322$, and XLNet achieving a F1 score of $0.7103$, while the domain-specific LEGAL-BERT model (pre-trained on legal texts) gives the best results, of $\approx 4\%$ increase over the XLNet model. These improvements in the metrics helped us choose LEGAL-BERT to be used as the base layer for our hierarchical transformer models. Bi-GRU over LEGAL-BERT was taken as the baseline model which shows a significant performance improvement over the previous models experimented on this dataset [5] as can be seen in the Table 3. With Bi-LSTM there is a slight improvement in the metric scores in both ILDC$_{single}$ and ILDC$_{multi}$. Adding dropouts over the Bi-LSTM layers results in a decrease in the performance in ILDC$_{single}$ for the same number of epochs ($= 6$) but an improvement in ILDC$_{multi}$. Since ILDC$_{single}$ is a small set as compared to ILDC$_{multi}$, adding dropouts slows down the model's ability to converge to the optimum boundary. Hence we trained this model for two more epochs to result in the performance improvement to $0.8084$ F1 score in ILDC$_{single}$ (Table 3). There was a marginal decrease in the metrics by using the dot-product attention while using the multi-head attention (with the query and key, value combination as shown in Table 3) resulting in slight performance improvements to $0.8070$ and $0.8125$ F1 scores in the test-set of ILDC$_{multi}$ and ILDC$_{single}$ respectively, for the hierarchical transformer model.

This shows that the dot-product attention and multi-head scaled dot-product attention mechanism used here do not improve the performance significantly. This can be pointed to the fact that the [CLS] embeddings used for the sequence models in hierarchical transformer, already contains the learnt representations from the internal multi-head attention function of the transformer architecture. To see if other novel attention mechanisms improve the performance of the hierarchical transformers is yet to be explored.

## 7. Conclusion

In this paper, we have explored the problem of decision classification of large unstructured and unannotated legal documents. We aim to formulate this problem as a decision prediction of legal case documents in real-life scenarios. To experiment with our models we used the ILDC dataset. We explored various state-of-the-art pre-trained language models (BERT [3], XLNet [17], LEGAL-BERT [2]), attention mechanisms, and sequence models (LSTM, GRU) for decision prediction tasks on the ILDC dataset. Based upon their performance we developed several baseline hierarchical domain-specific transformer models which improve significantly in the performance metrics of the previous models trained on the ILDC dataset. Our experiments show that LEGAL-BERT (a pre-trained domain-specific language model which is trained on the legal texts of Europe Union and United States court proceedings, each having their own specific legal terms, syntax and grammar), when fine-tuned on the legal case texts of The Supreme Court of India, adapts well to the grammar, lexicon, and syntax of Indian legal system. This finding shows that the domain-specific pre-trained language models can be adapted well to the

same domain with different language setting (syntax, grammar, lexicon). We aim to leverage this work on the prediction and classification of French legal cases in the future.

## Acknowledgments

This work was supported by the LawBot project, granted by ANR the French Agence Nationale de la Recherche.

## References

[1] J. A. Segal, Predicting supreme court cases probabilistically: The search and seizure cases, 1962-1981, American Political Science Review 78 (1984) 891–900. doi:10.2307/1955796.

[2] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, I. Androutsopoulos, LEGAL-BERT: the muppets straight out of law school, CoRR abs/2010.02559 (2020). URL: https://arxiv.org/abs/2010.02559. arXiv:2010.02559.

[3] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, CoRR abs/1810.04805 (2018). URL: http://arxiv.org/abs/1810.04805. arXiv:1810.04805.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, CoRR abs/1706.03762 (2017). URL: http://arxiv.org/abs/1706.03762. arXiv:1706.03762.

[5] V. Malik, R. Sanjay, S. K. Nigam, K. Ghosh, S. K. Guha, A. Bhattacharya, A. Modi, ILDC for CJPE: indian legal documents corpus for court judgmentprediction and explanation, CoRR abs/2105.13562 (2021). URL: https://arxiv.org/abs/2105.13562. arXiv:2105.13562.

[6] C. Xiao, H. Zhong, Z. Guo, C. Tu, Z. Liu, M. Sun, Y. Feng, X. Han, Z. Hu, H. Wang, J. Xu, CAIL2018: A large-scale legal dataset for judgment prediction, CoRR abs/1807.02478 (2018). URL: http://arxiv.org/abs/1807.02478. arXiv:1807.02478.

[7] I. Chalkidis, I. Androutsopoulos, N. Aletras, Neural legal judgment prediction in english, CoRR abs/1906.02059 (2019). URL: http://arxiv.org/abs/1906.02059. arXiv:1906.02059.

[8] H. Zhong, Z. Guo, C. Tu, C. Xiao, Z. Liu, M. Sun, Legal judgment prediction via topological learning, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 3540–3549. URL: https://aclanthology.org/D18-1390. doi:10.18653/v1/D18-1390.

[9] B. Luo, Y. Feng, J. Xu, X. Zhang, D. Zhao, Learning to predict charges for criminal cases with legal basis, CoRR abs/1707.09168 (2017). URL: http://arxiv.org/abs/1707.09168. arXiv:1707.09168.

[10] H. Zhong, Y. Wang, C. Tu, T. Zhang, Z. Liu, M. 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. URL: https://ojs.aaai.org/index.php/AAAI/article/view/5479.

[11] H. Chen, D. Cai, W. Dai, Z. Dai, Y. Ding, Charge-based prison term prediction with

deep gating network, CoRR abs/1908.11521 (2019). URL: http://arxiv.org/abs/1908.11521. arXiv:1908.11521.

[12] K. Cho, B. van Merrienboer, D. Bahdanau, Y. Bengio, On the properties of neural machine translation: Encoder-decoder approaches, CoRR abs/1409.1259 (2014). URL: http://arxiv.org/abs/1409.1259. arXiv:1409.1259.

[13] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, Neural Computation 9 (1997) 1735–1780. URL: https://doi.org/10.1162/neco.1997.9.8.1735. doi:10.1162/neco.1997.9.8.1735.

[14] M. Schuster, K. Paliwal, Bidirectional recurrent neural networks, IEEE Transactions on Signal Processing 45 (1997) 2673–2681. doi:10.1109/78.650093.

[15] D. Cer, Y. Yang, S. yi Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, Y.-H. Sung, B. Strope, R. Kurzweil, Universal sentence encoder, 2018. arXiv:1803.11175.

[16] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, CoRR abs/1908.10084 (2019). URL: http://arxiv.org/abs/1908.10084. arXiv:1908.10084.

[17] Z. Yang, Z. Dai, Y. Yang, J. G. Carbonell, R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, CoRR abs/1906.08237 (2019). URL: http://arxiv.org/abs/1906.08237. arXiv:1906.08237.

[18] R. Pappagari, P. Zelasko, J. Villalba, Y. Carmiel, N. Dehak, Hierarchical transformers for long document classification, CoRR abs/1910.10781 (2019). URL: http://arxiv.org/abs/1910.10781. arXiv:1910.10781.

[19] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. URL: http://arxiv.org/abs/1409.0473.

[20] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015. URL: http://arxiv.org/abs/1412.6980.