Don't Use a Cannon to Kill a Fly: An Efficient Cascading Pipeline for Long Documents

Zehua Li
zehuali@law.stanford.edu
Stanford University
Stanford, California, USA

Neel Guha
nguha@cs.stanford.edu
Stanford University
Stanford, California, USA

Julian Nyarko
jnyarko@law.stanford.edu
Stanford University
Stanford, California, USA

ABSTRACT

The computational cost of transformer-based models has a quadratic dependence on the length of the input sequence. This makes it challenging to deploy these models in domains in which long documents are especially lengthy, such as the legal domain. To address this issue, we propose a three-stage cascading approach for long document classification. We begin by filtering out likely irrelevant information with a lightweight logistic regression model before passing the more challenging inputs to the transformer-based model. We evaluate our approach using CUAD, a legal dataset with 510 manually-annotated, long contracts. We find that the cascading approach reduces training time by up to 80% while improving baseline performance. We hypothesize that the gains in performance stem from localizing the classification task of the transformer model to particularly difficult examples.

CCS CONCEPTS

• **Applied computing** → **Law**; *Document analysis*; • **Computing methodologies** → **Information extraction**.

KEYWORDS

Legal NLP, cascading classifiers, large language model

ACM Reference Format:
Zehua Li, Neel Guha, and Julian Nyarko. 2023. Don't Use a Cannon to Kill a Fly: An Efficient Cascading Pipeline for Long Documents. In *Nineteenth International Conference on Artificial Intelligence and Law (ICAIL 2023), June 19–23, 2023, Braga, Portugal.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3594536.3595142

1 INTRODUCTION

Legal long document analysis—the task of detecting whether a long legal document (e.g., a contract or lease) contains a particular type of clause and then analyzing it—is a common and important task for lawyers [21]. As part of due diligence, they may review a large set of contracts in order to determine the prevalence of clauses exposing a company to liability (e.g. *force majeure* provisions in the aftermath of COVID-19) [16]. In promoting access to justice, advocates may want to analyze the number of leases which contain oppressive or unenforceable terms [22]. Traditionally, this type of inquiry has been onerous, requiring individual lawyers to sift through many pages of documents in order to annotate and analyze all relevant provisions.

In recent years, large language models (LLMs)—powered by the transformer architecture [11]—have illustrated a remarkable capacity to match or exceed human performance on a variety of natural language tasks [5, 8, 11, 26, 30, 35]. Transformer models hold particular promise in the legal domain for two reasons. First, they can be trained without supervision, thus reducing the amount of task specific labeled data necessary to make good predictions. This is particularly beneficial in the legal context, where collecting task annotations can be prohibitively expensive [21]. Second, Transformer models rely on self-attention [36], allowing them to learn high quality *contextual* representations of input text. This produces higher performance on the types of lexically complex texts often found in the legal domain [6, 43].

However, what significantly detracts from their utility is that Transformer models are poorly suited for long document analysis. Because the cost of computing self-attention is quadratic in the length of the input sequence, most Transformer models are trained with a fixed context window, and can only process a limited number of tokens at a time. The most common window size for an open-source LLM—512 tokens—translates to approximately 1 page of text. In comparison, many legal documents can range between thirty to a hundred pages. The inability for practitioners to fit the legal document within the context window has thus significantly limited the impact Transformer models have had for legal applications. While some closed-source LLMs such as OpenAI's GPT-4 and Anthropic's Claude have a larger context window, processing an entire legal document in one request often remains computationally and/or economically infeasible [1, 31]. In addition, concerns for data privacy and confidentiality preclude most law firms from applying closed-source LLMs when serving and consulting their clients [14, 17].

Although the existing literature has proposed two types of approaches to address the obstacles for long documents analysis, both suffer from their own significant drawbacks when applied to the legal domain. In the first approach, researchers have investigated alternatives to the Transformer architecture, which may enable longer context windows [39, 42]. However, these methods only enable models with a context window up to 8,192 tokens (approximately 16 pages of text). These context windows are still insufficiently large for many legal applications, and they can sometimes perform worse than traditional Transformers [34]. Thus, under a second approach, a long document is "segmented" into a set of discrete chunks. A binary classification model is then applied to

ICAIL 2023, June 19–23, 2023, Braga, Portugal

ACM ISBN 979-8-4007-0197-9/23/06…$15.00
https://doi.org/10.1145/3594536.3595142

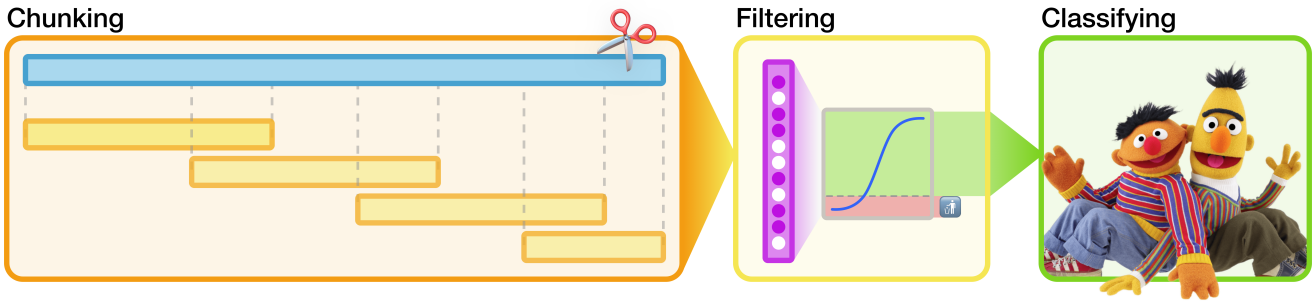

Figure 1: Our proposed workflow consists of three phases. In the first phase, we divide the input sequence into multiple chunks, where we use the training data to select the offset size between chunks. In the second phase, we use a logistic regression model to filter out chunks that are clearly irrelevant based on 100 bag-of-word features. In a last step, we feed the more challenging examples to Transformer-based models (such as BERT and ERNIE) for training and inference.

each chunk, which returns a positive label if the model predicts any of the segments to contain the provision in question [21]. However, this approach is computationally expensive, as practitioners must apply the Transformer model to each chunk in the document [21].

In this work, we present and study a novel workflow for performing legal long document analysis that is inspired by *cascading classifiers* [25, 37]. The intuition behind our approach is that much of the text contained in a long legal document is often unrelated to the task, and can thus be efficiently discarded without affecting performance. Our proposed workflow consists of three steps (Figure 1). First, we efficiently partition a long document into smaller sequences ("chunks"). Under minimal assumptions, our chunking algorithm guarantees that the target sequence is fully contained within at least one chunk, while minimizing the number of additional chunks not containing the target sequence. Second, we employ a computationally cheap classifier to efficiently discard chunks with a low probability of containing the target sequence. Finally, we apply a transformer model to the remaining chunks, and generate predictions over them. To demonstrate the utility of our approach, we consider the task of predicting the presence of individual clauses in long contracts[21]. We find that our pipeline improves performance while realizing significant gains in computational efficiency. Across three different clause-extraction tasks, we find that our method on average reduces the Transformer model's training time by 68.96%, while surpassing baseline performance by 1.97% ($p^* = 0.3$).

This paper proceeds as follows. In Section 2, we discuss related work and connect our workflow to prior approaches in computer vision. In Section 3, we provide an overview of our pipeline and discuss each of the three stages in greater detail. We present an empirical study of our approach in Section 4, and discuss broader implications in Section 5.

2 RELATED WORK

Large language models (LLMs)—also referred to as "foundation models" [4]—are million/billion parameter models trained on massive corpora of unlabelled text. Over the last several years, LLMs have demonstrated a consistent ability to outperform previous baselines on a wide array of natural language understanding tasks [8, 26, 35]. More recently, these models have been studied in the legal context, where their strong generalization capabilities mean that developers can achieve high performance with significantly smaller labeled datasets [6, 18, 43]. However, these models are typically trained with fixed context windows, and thus are incapable of processing longer input texts. As a result, their applications for legal tasks involving long legal documents (e.g. contracts or judicial opinions) are limited [6, 15, 21, 44].

In the context of law, approaches to extend the application of LLMs to long documents fall into two categories. In the first set of approaches, a document is divided into shorter, more manageable segments ("chunks"), which are sequentially processed by the model [2, 19, 21, 23]. The model's outputs over each chunk are then combined. For instance, [38] generate embeddings for each segment, and combine embeddings with a simple BiLSTM model. A downside of these approaches is that efficiency gains are often insufficient for practical use, because the number of chunks a model needs to process is very large. For instance, with a context window of 128 tokens, the approach used by [21] would split a 40,000 sub-word document into 312 segments to be fed into the LLM. For applications involving contracts—where only a subset of segments may actually be relevant for a given task—this is exceptionally wasteful, as the majority of segments are unrelated to the task [21].

In the second category of approaches, researchers have sought to develop alternatives or modifications to the traditional Transformer architecture which improve performance over longer sequences. Methods for reducing the quadratic cost of self-attention include adding sparsity [9] and using approximations for the self-attention computation [7, 24]. Other approaches explore alternatives to the traditional self-attention mechanism for the purpose of increasing the context window size [3, 10, 13, 33, 39, 42]. For instance, [41] extends [3] by pretraining on Chinese legal corpora with a context window of 4,096 tokens, while [28] extend a legal Longformer to a window size of 8,192 tokens. However, even larger context windows are often insufficiently large for legal documents, which can exceed one hundred pages. Additionally, for such long documents, models which apply some combination of local and global attention may be prohibitively expensive to employ. Finally, empirical results suggest

that these simplified versions of self-attention perform poorer than the standard Transformer architecture [34].

Our proposed pipeline is inspired by similar approaches for other machine learning tasks. Broadly, these approaches use a "cheap" classifier to filter out portions of the input that are unlikely to be relevant of the task, before applying the more "expensive" classifier to the filtered input. In computer vision, for instance, [37] developed classifiers that can quickly discard background areas of an image so that the downstream classifier can focus on the facial area. Applying the same strategy to NLP, [20] uses visual cues to sift through legal documents and isolate text of interest to feed the downstream LLM. This approach can also be found in the context of named entity recognition, where it is common for models to use simple heuristics to prune down the space of possible labels to a "candidate list," before using a more sophisticated model to pick a predicted label among the candidates [32, 40].

To our knowledge, two prior works propose cascading approaches to long document processing. First, [29] proposes TangoBERT, in which a smaller model is used to classify high-confidence inputs at inference time.[1] This approach differs from ours insofar as it focuses on reducing the computational cost only at the inference stage, while we also reduce the cost during training significantly. Additionally, [29] benchmarked on short text classification, where the length of most of its input texts is shorter than 256 tokens. Second, [12] proposes simultaneously training a "judge" and a "reasoner" model, where the judge model is used to select potentially relevant sentences from the document, and the reasoner model is tasked with generating a prediction over the concatenated selected sentences. [12] differs from us insofar as they focus on QA tasks, and both the judge and reasoner model are sophisticated Transformer based architectures. In contrast, our approach identifies relevant spans in longer documents by efficiently segmenting the document and utilizes a much simpler logistic regression classifier to filter out irrelevant input sequences. Lastly, unlike both [29] and [12], our specific focus lies on applications to the legal domain, and we provide some tailored suggestions to utilize the approach in this field. That said, in principle, the approaches provided by [29] and us are complementary, and we leave for future work how to combine them for further increases in computational efficiency.

3 METHODOLOGY

In our application, we treat the task as a binary classification task of smaller text chunks. Our choice is motivated by the fact that, in many real-world applications from the legal domain, readers of the documents under investigation are interested in the context under which the relevant text string appears. Providing them with additional contextual information thus poses advantages over contextualizing the task as a strict span identification task.

3.1 Data

We use CUAD for benchmarking. CUAD is a curated contracts dataset with 25 types of agreements, ranging from IP to non-compete agreements [21]. The contracts have been labeled using text spans for a set of 41 contract clauses typically of importance to a reader conducting contract review, such as effective dates and renewal terms. Among the different clauses, we focus on *Audit Rights*, *Cap on Liability* and *License Grant*. We select these clauses because they are of significant length, of substantive interest to a potential reader and non-trivial to identify.

[1]The study was developed in parallel to our own work.

Table 1: Chunk statistics after applying our efficient chunking algorithm. A chunk that contains over 50% of any given labeled span is considered positive. The table lists the number of positive/negative chunks for each of our three clause types, separately for the training set and the test set.

Clause Type	Training		Testing	
	#positive	#negative	#positive	#negative
Audit Rights	2,402	66,031	470	14,215
License Grant	3,271	65,162	728	13,957
Cap on Liability	1,657	37,356	319	8,060

3.2 Classification Pipeline

Our classification pipeline consists of three steps. First, we divide the long documents into smaller chunks, ensuring that each labeled sequence is fully contained in at least one chunk. In a second step, we feed the chunks into a logistic classifier that uses as input the 100 words with the highest information gains and predicts whether the chunks contain the labeled sequence. In a last step, we feed the chunks with a sufficiently high predicted probability of containing the relevant label into a RoBERTa model to create our final predictions. We describe each step in more detail below.

3.2.1 Chunking. A traditional approach to working with long input strings is the "sliding-window" approach. In it, inputs are generated by sliding a fixed-size window over the long document by a predetermined stride. However, this approach does not take advantage of the known distribution of the labels' length for a given task. For tasks in which labeled spans are short, this approach generates a redundant amount of inputs. For tasks in which spans are long, a long stride creates the risk that none of the populated chunks fully contains a span of interest.

In contrast, we propose a chunking process that utilizes information on the distribution of the lengths of the labeled spans in the training data and finds an efficient way to segment the document, subject to the constraint that at least one chunk must fully contain each of the labeled spans. More formally, let W be window or chunk size and S be the stride, i.e. the number of words by which the window is moved. Further, T denotes the length of the text span of interest. Then our chunking process chooses S such that

$$\begin{aligned} \max \quad & S \\ \text{s.t.} \quad & S < W - T_{max} \end{aligned} \tag{1}$$

For every labeled span in the training set, as long as the stride size is smaller than $W - T_{max}$, there exists one chunk where the text is fully contained. In application, we set a lower bound of S to 64 tokens to avoid generating too many chunks. For reference, only 0.20% of the labeled spans in our experiments are longer than $W - 64$ tokens.

3.2.2 Filtering. Next, we featurize the text using a bag-of-words model and feed a subset of the input features into a computationally cheap classification algorithm. Among the range of plausible candidates, we opt for a logistic classifier for its simplicity and computational efficiency. For each input chunk, the logistic classifier estimates a predicted probability p that the chunk contains the labeled sequence.

A key parameter of our classification pipeline is the threshold parameter on the predicted probability, which we denote p^*. Effectively, this threshold parameter allows the user to determine how many chunks are passed on to the final classification stage. A low p^* (permissive threshold) ensures that many chunks are filtered to the last classification step, allowing the (more precise) Transformer-based model to correct mistakes by the (less precise) logistic classifier. At the same time, a low p^* increases the computational cost of the classification pipeline. In contrast, a high p^* (impermissive threshold) filters out more chunks, increasing computational efficiency at the potential cost of reduced performance.

Because the appropriate threshold p^* can vary from one application to the next, based on the available computational resources and the tolerance for mistakes, it is recommended to choose p^* after examination of different performance criteria. In most legal applications, computational outputs are subject to final review. In this setting, the costs of a false negative far outweigh the costs of a false positive, because reviewing for false negatives requires re-reading the entire long document. In this setting characterized by risk aversion, it appears particularly recommended to focus on recall at the initial classification stage.

Since logistic regression tends to overfit on training data when the number of features is large, we select as inputs the top 100 features that have the highest information gains, meaning they are statistically most indicative of the label of a chunk. We then feed the text chunk to the classifier using the 100-token bag-of-word representation. During training, positive examples are given 1,000 times the weight of negative examples to ensure that the second-stage model only filters out examples that are clearly negative.

3.2.3 RoBERTa Classification. In a last step, we use a computationally costly but more accurate classifier to generate the final predictions. Once again, there are a range of classifiers to choose from, and we opt for a finetuned RoBERTa classifier.

RoBERTa is a large language model that is structurally the same as BERT but is trained with a dataset 10 times larger than BERT's corpus.[27] We finetune the model using training examples passed down from the filtering stage, which include the true positives, false positives, and false negatives. We include examples the logistic classifier mistakenly labeled as negative to increase the number of positive examples in RoBERTa's training dataset.

4 RESULTS

We test the proposed approach against three clauses in CUAD (*Audit Rights*, *Cap on Liability*, and *License Grant*). In the following section, we investigate how the pipeline performs, stage by stage, focusing on the example of *Audit Rights*.

Table 2: The performance of the logistic classification model on Audit Rights given different predicted probability threshold p^*. The Precision and F1 columns are colored gray because in most legal applications, the primary focus of the user should be on Recall and % passed to RoBERTa under the different thresholds p^*.

Threshold p^*	Precision	Recall	F1	% passed to RoBERTa
0.1	12.79%	97.02%	22.60%	24.28%
0.2	14.29%	97.02%	24.90%	21.74%
0.3	15.40%	97.02%	26.57%	20.17%
0.4	16.41%	96.81%	28.06%	18.88%
0.5	17.25%	95.96%	29.25%	17.80%
0.6	18.14%	95.96%	30.51%	16.93%
0.7	19.14%	95.74%	31.90%	16.01%
0.8	20.40%	94.89%	33.58%	14.89%
0.9	22.19%	93.40%	35.87%	13.47%

4.1 Chunking

The longest labeled span for Audit Rights has a length T_{max} of 465 tokens, while the shortest has a length T_{min} of 2. We therefore set the stride to $max(64, W - T_{max} + 1) = 64$ to avoid generating an excessive amount of chunks. The chunking process generates 83, 118 chunks across the entire dataset. The statistics are depicted in Table 1.

4.2 Filtering

The logistic classifier, despite its relative simplicity, performs well on the task of filtering out obviously negative inputs. As seen in Table 2, the model manages to trim out a significant portion of input chunks while maintaining a high recall. It can filter out 79.83% of the inputs with a recall of 97.02%, or maintain a 94.89% recall while passing down only 14.89% of the inputs. That is because, as shown in Figure 2, the vast majority of the inputs have a predicted probability lower than 0.1. Since training of the RoBERTa model without stopping conditions is of linear time complexity with regards to the number of training examples, using the second-stage logistic classification model filters out simple negative examples. The user can decide on a predicted probability threshold p* based on their risk tolerance, computational resources, the model's recall and the percentage of inputs it passed down to RoBERTa.

For the purpose of demonstrating that the pipeline performs well across different thresholds, rather than choosing a specific threshold, we evaluate it on multiple thresholds.

4.3 RoBERTa Classification

The performance of the RoBERTa classification model is depicted in the second row of Figure 3. The AUROCs are high across all filtering thresholds, with a slight decrease towards higher thresholds, suggesting that the classification becomes more challenging as there are fewer inputs that are obviously negative.

Assessing the entire pipeline at a threshold of $p^* = 0.3$, the three-stage process achieves a higher AUROC (97.28%) compared to that of the baseline RoBERTa model (92.41%), while reducing the pipeline's total training time by 82.67%. Our proposed method also achieves an F1 score (79.22%) on par with the baseline model's

Table 3: Performance and the training/inference time of the entire pipeline using different filtering thresholds p^*. In the baseline where $p^* = 0.0$, every input is passed down to the second-stage RoBERTa-base model for training and evaluation. The pipelines are allowed to utilize 8 Intel Xeon Platinum 8259CL CPUs and 3 Nvidia Tesla T4 GPUs.

Clause	p^*	Second Stage (LR)		Third Stage (RoBERTa)			Entire Pipeline			
		Recall	%examples	Recall	F1	$Time_T$	F1	AUROC	$Time_T$	$Time_I$
Audit Rights	0.0	—	100.00%	79.79%	78.95%	100.00%	78.95%	92.41%	100.00%	100.00%
	0.1	97.02%	24.28%	78.73%	76.71%	20.24%	75.58%	96.86%	22.02%	23.57%
	0.2	97.02%	21.74%	88.16%	79.53%	17.47%	78.44%	97.25%	19.24%	21.29%
	0.3	97.02%	20.17%	84.87%	80.37%	15.56%	79.22%	97.28%	17.33%	20.03%
	0.4	96.81%	18.88%	87.25%	82.97%	15.11%	81.69%	97.29%	16.88%	18.79%
	0.5	95.96%	17.80%	86.70%	78.75%	13.92%	77.27%	96.78%	15.70%	17.75%
	0.6	95.96%	16.93%	86.70%	82.75%	13.19%	81.12%	96.97%	14.97%	16.90%
	0.7	95.74%	16.01%	86.67%	79.67%	12.02%	78.08%	96.81%	13.80%	15.93%
	0.8	94.89%	14.89%	88.57%	80.78%	10.72%	78.84%	96.26%	12.50%	14.80%
	0.9	93.40%	13.47%	85.65%	80.26%	9.79%	77.69%	95.63%	11.58%	13.41%
License Grant	0.0	—	100.00%	77.34%	76.65%	100.00%	77.34%	95.05%	100.00%	100.00%
	0.1	96.70%	36.71%	80.97%	79.94%	33.66%	78.62%	96.12%	35.40%	52.45%
	0.2	95.88%	33.79%	77.94%	79.13%	30.44%	77.44%	95.82%	32.18%	48.22%
	0.3	94.78%	31.80%	83.04%	81.57%	27.69%	79.42%	95.12%	29.42%	45.97%
	0.4	94.23%	30.25%	78.86%	79.33%	26.25%	76.96%	94.97%	27.98%	42.38%
	0.5	93.96%	28.93%	80.12%	80.71%	24.88%	78.17%	95.00%	26.62%	39.91%
	0.6	93.41%	27.85%	81.18%	79.20%	23.74%	76.56%	94.63%	25.47%	38.46%
	0.7	92.03%	26.52%	81.49%	80.47%	21.62%	77.17%	93.82%	23.37%	35.49%
	0.8	91.07%	24.93%	82.96%	81.60%	20.48%	77.85%	93.50%	22.21%	33.15%
	0.9	89.56%	22.48%	83.31%	81.32%	19.77%	77.00%	92.65%	21.50%	29.62%
Cap on Liability	0.0	—	100.00%	74.61%	76.16%	100.00%	76.16%	94.29%	100.00%	100.00%
	0.1	96.87%	54.62%	77.35%	78.23%	52.43%	76.97%	92.07%	55.06%	35.75%
	0.2	96.87%	49.42%	76.70%	79.00%	47.49%	77.70%	94.94%	50.15%	32.71%
	0.3	96.55%	45.86%	78.57%	78.70%	43.67%	77.32%	95.25%	46.35%	30.85%
	0.4	95.92%	43.22%	78.43%	78.30%	41.85%	76.68%	94.83%	44.53%	29.30%
	0.5	95.61%	40.67%	76.07%	76.69%	38.48%	74.96%	94.82%	41.16%	28.18%
	0.6	95.30%	38.38%	76.97%	79.86%	35.50%	77.87%	94.94%	38.20%	26.88%
	0.7	93.73%	36.11%	79.26%	79.80%	34.40%	77.20%	94.43%	37.10%	25.58%
	0.8	93.10%	33.24%	84.18%	83.19%	31.81%	80.26%	94.09%	34.53%	24.15%
	0.9	92.16%	29.71%	81.97%	81.14%	27.40%	77.87%	93.97%	30.10%	23.79%

78.95%. We surmise that is because, as the logistic regression model filters out most of the examples at the training stage with high confidence, the RoBERTa model can focus on discerning a narrower band of cases the upstream model deems challenging, therefore achieving a higher overall AUROC with much fewer training examples.

4.4 Additional Experiments

We evaluate the proposed approach on two other legal clauses in CUAD, *Cap on Liability* and *License Grant*. The performance can be found in Table 3. In both cases, the proposed pipeline achieves an AUROC score that is on par with or surpassing the baseline performance, suggesting the pipeline is adaptable to different tasks. For *License Grant*, at a threshold of $p^* = 0.3$, the pipeline achieves an AUROC score of 95.12% compared to the baseline model's 95.05%. For *Cap on Liability*, the pipeline surpasses the baseline's 94.29% AUROC, demonstrating a better ability at discerning positive chunks. Again, we note that the higher performance might be a consequence of the more focused training task that is centered around distinguishing difficult examples. The pipeline achieves high performance while significantly reducing the RoBERTa training time by at least 47.6%.

5 DISCUSSION

We proposed a three-stage cascading pipeline for long legal document classification. In a legal document classification benchmark, the filter based on a logistic regression classifier accurately removes a sizeable proportion of negative inputs while feeding a smaller portion of challenging examples to the Transformer-based classifier. The efficient cascading pipeline achieves performance surpassing the an alternative approach without a filter. Our suggested pipeline is simple to integrate, significantly decreases the amount of time spent training the Transformer-based model and speeds up long-input inference.

Unlike other approaches that implement the efficiency-improving measures into the LLM architecture (such as BigBird [42]), the proposed three-stage pipeline allows researchers to balance overall efficiency against predictive quality by adjusting the proportion of negative examples to filter at the second stage. Our approach can easily be integrated with existing classification systems. For instance, users can replace the third-stage RoBERTa base model with a Longformer model that supports inputs as long as 4,096 tokens. The longer input sequence, together with the dynamic chunking, increase the maximally allowed stride to $4096 - t_{max} + 1$, potentially improving the pipeline efficiency even further.

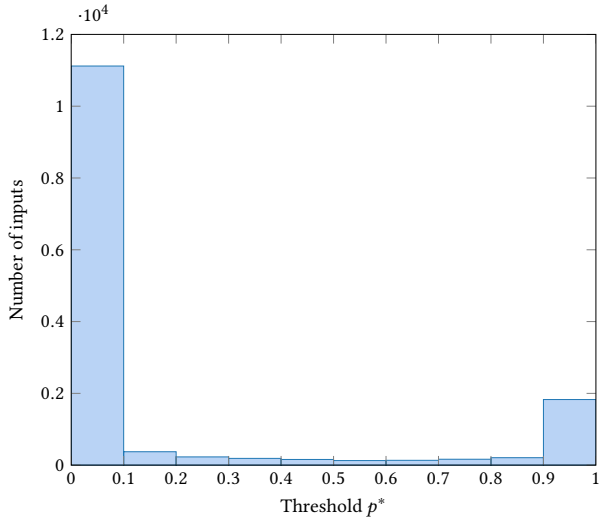

Figure 2: The distribution of the predicted probabilities for the Audit Rights test set. The vast majority of the inputs (75.6%) have a predicted probability lower than 0.1.

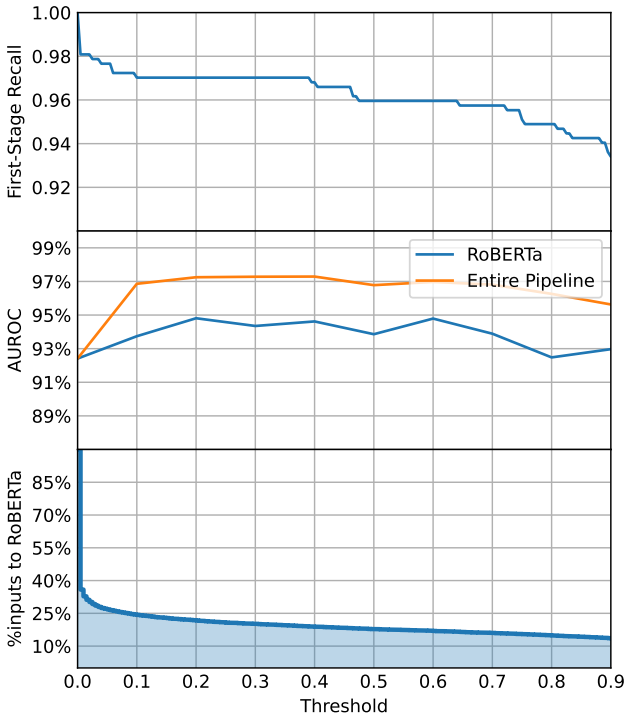

Figure 3: The performance of the pipeline on Audit Rights at different stages.

Inputs that are clearly irrelevant to the task should not take the same amount of floating point operations to classify as more challenging inputs do. The model should also spend less time and energy finetuning on simple examples. The proposed three-stage approach thus complements the latest development in LLMs by directing the Transformer-based models to more challenging inputs and shortening the finetuning process.

As this paper primarily focuses on the three-stage pipeline's application in legal document classification, additional evaluations are needed to assess the applicability and utility of the cascading approach to the broader set of tasks involving long document classification and information retrieval. We also do not explore the integration of the cascading pipeline with other efficiency-improving measures. We look forward to future research on cascading LLM pipelines that dynamically target the training and inferring efforts to the more challenging examples.

REFERENCES

[1] Anthropic AI. 2023. Introducing Claude. https://perma.cc/LLR5-YZCC.
[2] Purbid Bambroo and Aditi Awasthi. 2021. Legaldb: Long distilbert for legal document classification. In *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*. IEEE, 1–4.
[3] Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150* (2020).
[4] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2021. On the Opportunities and Risks of Foundation Models. https://doi.org/10.48550/ARXIV.2108.07258
[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.
[6] Ilias Chalkidis, Abhik Jana, Dirk Hartung, Michael Bommarito, Ion Androutsopoulos, Daniel Martin Katz, and Nikolaos Aletras. 2021. Lexglue: A benchmark dataset for legal language understanding in English. *arXiv preprint arXiv:2110.00976* (2021).
[7] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. 2020. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794* (2020).
[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311* (2022).
[9] Xiang Dai, Ilias Chalkidis, Sune Darkner, and Desmond Elliott. 2022. Revisiting transformer-based models for long document classification. *arXiv preprint arXiv:2204.06683* (2022).
[10] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860* (2019).
[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/ARXIV.1810.04805
[12] Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang. 2020. Cogltx: Applying bert to long texts. *Advances in Neural Information Processing Systems* 33 (2020),

12792–12804.

[13] Siyu Ding, Junyuan Shang, Shuohuan Wang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. ERNIE-Doc: A retrospective long-document modeling transformer. *arXiv preprint arXiv:2012.15688* (2020).

[14] Emily Dreibelbis. 2023. Samsung Software Engineers Busted for Pasting Proprietary Code Into ChatGPT. https://www.pcmag.com/news/samsung-software-engineers-busted-for-pasting-proprietary-code-into-chatgpt. https://www.pcmag.com/news/samsung-software-engineers-busted-for-pasting-proprietary-code-into-chatgpt

[15] Mahmoud El-Haj, Nadhem Zmandar, Paul Rayson, Marina Litvak, Nikiforos Pittaras, George Giannakopoulos, Aris Kosmopoulos, Blanca Carbajo-Coronado, Antonio Moreno-Sandoval, et al. 2022. The Financial Narrative Summarisation Shared Task (FNS 2022). In *Proceedings of the 4th Financial Narrative Processing Workshop@ LREC2022*. 43–52.

[16] Ryan Franklin and Nicholas Wind. 2022. Force Majeure Clauses in the Aftermath of the Covid-19 Pandemic and the Implications for Government Entities. https://www.americanbar.org/groups/government_public/publications/pass-it-on/spring-2022/spring22-franklin-wind-forcemajeure/.

[17] Karla Grossenbacher. 2023. Employers Should Consider These Risks When Employees Use ChatGPT. https://news.bloomberglaw.com/us-law-week/employers-should-consider-these-risks-when-employees-use-chatgpt.

[18] Neel Guha, Daniel E Ho, Julian Nyarko, and Christopher Ré. 2022. LegalBench: Prototyping a Collaborative Benchmark for Legal Reasoning. *arXiv preprint arXiv:2209.06120* (2022).

[19] Marti A Hearst. 1994. Multi-paragraph segmentation of expository text. *arXiv preprint cmp-lg/9406037* (1994).

[20] Allison Hegel, Marina Shah, Genevieve Peaslee, Brendan Roof, and Emad Elwany. 2021. The Law of Large Documents: Understanding the Structure of Legal Contracts Using Visual Cues. (2021). https://doi.org/10.48550/ARXIV.2107.08128

[21] Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. 2021. CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review. https://doi.org/10.48550/ARXIV.2103.06268

[22] David A Hoffman and Anton Strezhnev. 2022. Leases as forms. *Journal of Empirical Legal Studies* 19, 1 (2022), 90–134.

[23] Maor Ivgi, Uri Shaham, and Jonathan Berant. 2023. Efficient long-text understanding with short-text models. *Transactions of the Association for Computational Linguistics* 11 (2023), 284–299.

[24] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451* (2020).

[25] Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing Neural Predictions. https://doi.org/10.48550/ARXIV.1606.04155

[26] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2022. Holistic Evaluation of Language Models. https://doi.org/10.48550/ARXIV.2211.09110

[27] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://doi.org/10.48550/ARXIV.1907.11692

[28] Dimitris Mamakas, Petros Tsotsi, Ion Androutsopoulos, and Ilias Chalkidis. 2022. Processing Long Legal Documents with Pre-trained Transformers: Modding LegalBERT and Longformer. *arXiv preprint arXiv:2211.00974* (2022).

[29] Jonathan Mamou, Oren Pereg, Moshe Wasserblat, and Roy Schwartz. 2022. TangoBERT: Reducing Inference Cost by using Cascaded Architecture. https://doi.org/10.48550/ARXIV.2204.06271

[30] OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023). arXiv:2303.08774 [cs.CL]

[31] OpenAI. 2023. Product Specification for GPT-4. https://perma.cc/53LN-APQT.

[32] Laurel Orr, Megan Leszczynski, Simran Arora, Sen Wu, Neel Guha, Xiao Ling, and Christopher Re. 2020. Bootleg: Chasing the tail with self-supervised named entity disambiguation. *arXiv preprint arXiv:2010.10363* (2020).

[33] Raghavendra Pappagari, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. 2019. Hierarchical transformers for long document classification. In *2019 IEEE automatic speech recognition and understanding workshop (ASRU)*. IEEE, 838–844.

[34] Hyunji Hayley Park, Yogarshi Vyas, and Kashif Shah. 2022. Efficient classification of long documents using transformers. *arXiv preprint arXiv:2203.11258* (2022).

[35] Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model. *arXiv preprint arXiv:2211.05100* (2022).

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. https://doi.org/10.48550/ARXIV.1706.03762

[37] Paul Viola and Michael J Jones. 2004. Robust real-time face detection. *International journal of computer vision* 57, 2 (2004), 137–154.

[38] Lulu Wan, George Papageorgiou, Michael Seddon, and Mirko Bernardoni. 2019. Long-length legal document classification. *arXiv preprint arXiv:1912.06905* (2019).

[39] Chuhan Wu, Fangzhao Wu, Tao Qi, Yongfeng Huang, and Xing Xie. 2021. Fastformer: Additive Attention Can Be All You Need. https://doi.org/10.48550/ARXIV.2108.09084

[40] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Zero-shot Entity Linking with Dense Entity Retrieval. In *EMNLP*.

[41] Chaojun Xiao, Xueyu Hu, Zhiyuan Liu, Cunchao Tu, and Maosong Sun. 2021. Lawformer: A Pre-trained Language Model for Chinese Legal Long Documents. https://doi.org/10.48550/ARXIV.2105.03887

[42] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems* 33 (2020), 17283–17297.

[43] Lucia Zheng, Neel Guha, Brandon R Anderson, Peter Henderson, and Daniel E Ho. 2021. When does pretraining help? assessing self-supervised learning for law and the casehold dataset of 53,000+ legal holdings. In *Proceedings of the eighteenth international conference on artificial intelligence and law*. 159–168.

[44] Nadhem Zmandar, Mahmoud El-Haj, Paul Rayson, Marina Litvak, Geroge Giannakopoulos, Nikiforos Pittaras, et al. 2021. The financial narrative summarisation shared task fns 2021. In *Proceedings of the 3rd Financial Narrative Processing Workshop*. 120–125.