Retrieval-Augmented Generation Inspired Long Document Classification

Anonymous ACL submission

Abstract

While being proven to be effective across nearly all natural language processing tasks, transformer-based models have several obvious limitations. Amongst them, arguably the most significant one is the quadratic complexity – both in time and space – of the vanilla self-attention mechanism. As a result, most existing pre-trained language models, such as BERT, have a fixed maximum context window. This potentially creates a mismatch between the context window size and the data applied to fine-tuning it. This gives rise to the study of long document classification – the task to optimize performance when the length of the input document exceeds the model's maximum token. Inspired by retrieval-augmented generation techniques used by large language models in recent years, we propose a method that uses similar techniques to retrieve the most relevant sections of a long document, which is then fed into a traditional transformer-based model. By testing on four standard long document classification datasets, we show that our proposed method on average outperforms all baselines, including both transformer and non-transformer based models.

1 Introduction

While transformer-based models have demonstrated their efficacy across nearly all natural language processing tasks in recent years, they come with several caveats. Arguably the most significant limitation is the quadratic complexity inside the computation of the vanilla self-attention mechanism in each of the encoder (and decoder if applicable) layers of the model. As a result, most existing pre-trained language models have a fixed maximum context window. For example, BERT (Devlin et al., 2019) has a context window of 512, whereas encoder-decoder models, such as BART, (Lewis et al., 2020) have a window of 1,024. While some later models such as T5 (Raffel et al., 2020) removed this limit theoretically by using techniques such as relative positional embeddings (Shaw et al., 2018), in practice processing arbitrarily long inputs would quickly lead to out-of-memory errors. However, in real-life applications, especially in certain domains such as legal text, it is often desirable for models to be able to handle documents far longer than the supported context window. This gives rise to the study of long document classification, the research on optimizing model performance on inputs which exceed the token limit of a model.

Existing approaches to long document classification can be categorized into two directions. The first is to extend the context window of pre-trained language models, through reducing the complexity (and hence memory requirement) of the self-attention mechanism. Numerous types of sparse attention have be proposed over the years, resulting in models such as Longformer (Beltagy et al., 2020), which can handle documents up to 4,096-token sequences. However, these models still have a theoretical maximum context window, which in recent years appears dwarfed by large language models' (LLMs) ability to process up to 1 million tokens at once.[1]

The other direction is to reduce the length of the input document so as to fit onto a smaller window of a pre-trained language model. Work such as CogLTX (Ding et al., 2020) aims to find the "most significant" chunk or section of the long text, and proceeds to fine-tune a model with the shortened text. The challenge of this lies within the fact that there is no well-defined concept of significance, and no labelled data to train on, whence this must be done in an unsupervised or self-supervised manner, greatly hampering the efficacy.

However, this second approach bores a striking resemblance with retrieval-augmented generation

[1] https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024/

(RAG), a commonly used practice by LLMs to combat hallucination. In real-world applications, LLMs are often tasked with answering a query based on a database or collection of documents, which can easily be hundreds of thousands of pages containing company policies, legal or medical advices, etc. Thus, one fundamental task of RAG is to retrieve the most relevant document(s) within the large collection, which can then be appended to the query and served as the context to the LLM. This is a very active area of research, with focuses on methods such as directly training a specialized retriever model (Trivedi et al., 2023), applying a two-stage retrieve-then-rerank framework (Sun et al., 2023a) and query rewriting (Gao et al., 2023a) just to name a few.

Drawing inspirations from RAG, we propose a method for long document classification by training a reranker model which assigns a numerical score to every sentence in a long document to represent its significance to the classification. The model learns from a teacher LLM, by which we leverage its naturally longer context window in order to determine the most significant sections of the document. After training, the reranker model is applied on the entire dataset to obtain a shortened version of the long document and fine-tune a standard pre-trained language model as usual. Our contributions are summarized as follows:

- To the best of our knowledge, this is the first work to link long document classification with RAG, and the first to apply related techniques into long document classification;
- Testing on four standard long text classification datasets, we empirically show that our method outperforms all baselines on three datasets, while remaining competitive on the remaining one;
- We provide extensive discussion on the results, including ablation studies on the efficacy of the individual components.

The rest of the paper is structured as follows. Section 2 presents some background information about long-context transformers and RAG. We illustrate our proposed model architecture in Section 3, followed by supporting experimental designs in Section 4. Empirical results and analysis, as well as ablation studies, are documented in Section 5. Limitations are discussed in Section 6 and we conclude our work in Section 7 with some future directions.

2 Related Work

2.1 Transformers for Long Documents

Given a long document and a pre-trained language model with a shorter context window, there are two natural approaches to address the problem of long document: either make the context window longer or make the document shorter.

Most work, which focuses on making the context windows longer, revolved around reducing the computation complexity of the attention mechanism. Wang et al. (2020) argued that self-attention matrices were theoretically and empirically low-ranked, and proposed Linformer by using a linear complexity self-attention mechanism. Parameters sharing were explored to further increase the inference speed. Beltagy et al. (2020) introduced Longformer, which is one of the most widely adopted model used for long documents. They replaced the full self-attention with various types of sliding window attentions, combining local and global information to achieve state-of-the-art results on various standard long document datasets. However, despite their success, it was suggested in Narang et al. (2021) that most modifications to the vanilla transformer architecture did not meaningfully improve performance, and these apparent improvements were purely based on implementation details. Intuitively, this is in line with the fact that there should always a trade-off between complexity and performance.

Other work such as CogLTX (Ding et al., 2020) chunks a long document and trains a separate BERT model to judge the relevancy of each block and select the most relevant chunks. They overcome the lack of relevancy labels in the input text by inferring the information through intervention – removing a block from the entire document and using a separate BERT model to test whether it is indispensable. Our work pursues this line of thought and is very similar in principle to CogLTX. However, we propose a different method to obtain relevancy labels, inspired by LLMs and RAG.

In more recent years, LLMs, such as the GPT series[2], PaLM2/Gemini[3], and Llama2/3[4], have demonstrated superior capabilities across multiple tasks. While their significantly longer context windows, ranging from 4,096 up to one million along

[2]https://openai.com/index/gpt-4-research/

[3]https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024

[4]https://llama.meta.com/llama3/

with unparalleled natural language understanding abilities, seem to trivialize the study of long document classification at first glance, various research has proven otherwise. Sun et al. (2023b) argued that LLMs inherently underperformed a fine-tuned model on most text classification tasks, since most recent efforts have been made to improve their logical reasoning abilities while neglecting the ability to understand intricate linguistic phenomena in text (such as concession, negation etc). It is also hindered by the token limitation in commonly used techniques such as in-context learning (Dong et al., 2023). Furthermore, Liu et al. (2023) showed that although LLMs handled long contexts, much of the information was lost in the middle, and the generated outputs were often only based on the start and end of the provided context. In real-world applications, there are also practical considerations when deploying an LLM-based solution, in terms of cost (either through acquiring GPUs or by directly calling an API), latency, and data security. These evidences validate the importance of studying long document classification; in particular, in view of the practical considerations of LLMs, we aim to develop a system that is non-reliant of LLMs during inference.

2.2 Retrieval-Augmented Generation for Large Language Models

Despite the immense success of LLMs, all such generative models suffer from some degree of hallucination, misinformation or outdated knowledge (Huang et al., 2023), often resulting in some bemusing results.[5,6] Over the years, RAG has become one of the standard measures to counter these issues.

The principle of RAG is simple: given a query, instead of relying on the implicit knowledge that an LLM obtained during pre-training, one attempts to "override" this knowledge by explicitly providing the context for the LLM to base its response on. This has additional benefits of allowing LLMs to answer queries on latest news or proprietary information without the need to fine-tune the entire model. In practice, however, providing the most relevant context can be a challenge *per se*, especially when the knowledge base contains hundreds of thousands of documents. Thus, how to select the most relevant document(s) from a large knowledge base is an important focus of RAG research and application.

[5]https://www.bbc.com/news/business-64576225

[6]https://www.bbc.com/news/articles/cv2xx1xe2evo

The standard pipeline for document selection is often two-staged: indexing and retrieval. Indexing is simply the process of chunking a large database into smaller, digestible chunks; retrieval is the design of the algorithm used to select the top-k chunks given a user query, usually via some similarity scores between vector embeddings of the input and the database. In more advanced setups, indexing can be done alongside extraction of keywords or entities[7], which also serves as a first-pass document filter during retrieval. After retrieval, a specialized reranker can also be implemented to refine the retrieval results (Zhang et al., 2023). RAG is a vast and rapidly expanding area. While we have barely scratched the surface here, interested readers can refer to Gao et al. (2023b) for a more detailed survey on the topic.

Our motivation comes from the similarities between RAG and long document classification. In essence, the aforementioned work such as CogLTX acts as the equivalence of a retriever, operating on the long document itself with the aim to select the most relevant sections or sentences within the document. Inspired by this connection, we propose a method that leverages both LLMs and reranking models, by applying them to the context of long document classification.

3 Problem Statement & Methodology

3.1 Problem Statement

Long text classification is a special case of general text classification. Given a set of text inputs $\mathcal{X}$ and the corresponding labels $\mathcal{Y}$, text classification simply aims to compute a mapping

$$f_M : \mathcal{X} \to \mathcal{Y} \, ,$$

using a model M. Assuming that M is a transformer-based model with maximum context window of L tokens, long text classification assumes that a non-trivial proportion of the data $\mathcal{X}$ contains more than L tokens. If all elements in labels set $\mathcal{Y}$ contains only one label, we call this multiclass classification; if the elements is a list of labels (i.e. $y = [y_1, y_2, \ldots, y_n]$), we call this multilabel classification.

3.2 Overview

Our proposed method is illustrated in Figure 1. During training (c.f. Figure 1(a)), we first leverage an

[7]https://docs.llamaindex.ai/en/stable/module_guides/indexing/metadata_extrac

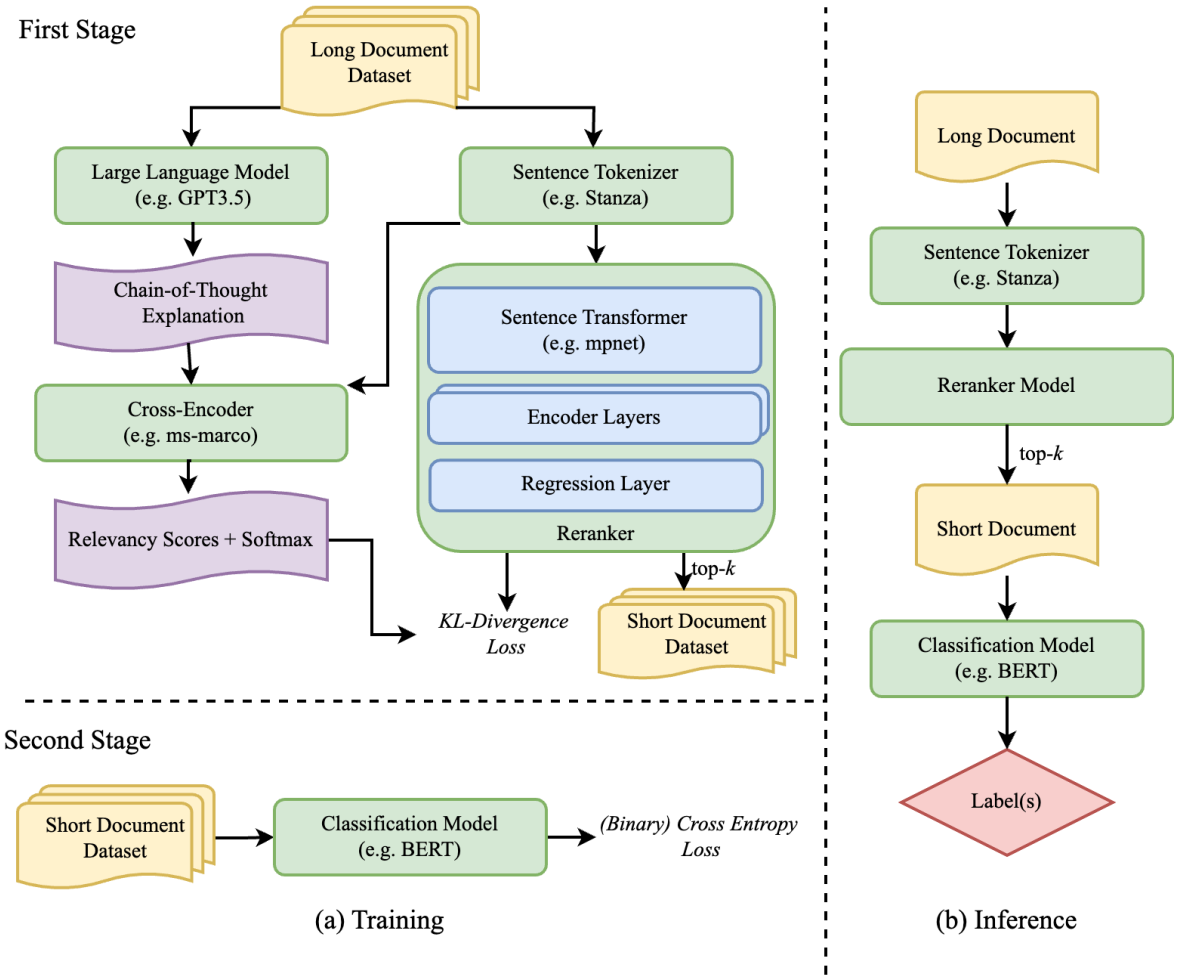

Figure 1: Our proposed method. Figure 1(a) depicts the two-staged training pipeline. A reranker model is first trained on relevancy scores obtained via an LLM; then a classifier model is trained on a shortened version of the original dataset. Figure 1(b) depicts the inference pipeline, where a long document is passed through the trained reranker to obtain a shortened representation before passing through the trained classifier.

LLM to obtain relevancy scores for each sentence in a long document. A specialized reranker model is trained based on these generated labels. Once the reranker is trained, it is used to generate a short document dataset, which is then used to train a classification model in a standard fine-tuning fashion. During inference (c.f. Figure 1(b)), the reranker takes in a long document and predicts the most relevant sentences, which is then concatenated and fed into the classifier to obtain the final output, i.e. labels.

We break down each step of our method in more details below.

3.3 Relevancy Scores

Given a long document $x = [x_1; \ldots; x_n]$, where x_i denotes the i-th sentence obtained via any open-source tools, and its corresponding labels, we leverage a "reverse" chain-of-thought prompt and prompt an LLM to generate an explanation for each provided label

$$\text{Reason}_x = \text{LLM}(\text{prompt}, x) \ .$$

To make the downstream training easier, we further engineer the prompt to explicitly ask the LLM to give references to the original text whenever possible. The exact prompts used for each dataset can be found in the Appendix.

Using the output of the LLM, we pass it through a pre-trained cross-encoder CE along with each sentence in the original document, thus obtaining a relevancy score that represents the importance of the sentence to explain the classification labels:

$$\text{Relevancy}_x = \text{softmax}(\{CE(\text{Reason}_x; x_i) \text{ for } x_i \text{ in } x\})$$

Softmax turns the scores into a distribution, the reason for which will become immediately apparently

Dataset	Label Type	Classes	Size	# BERT Tokens	# Sentences
Hyperpartisan	Multiclass	2	645	744.18 ± 677.87	30.51 ± 28.64
20 News Groups	Multiclass	20	18,846	368.83 ± 783.84	21.01 ± 36.18
EURLEX	Multilabel	4,271	57,000	707.99 ± 538.69	7.77 ± 8.344
Book Summaries	Multilabel	227	12,788	574.31 ± 659.56	20.74 ± 25.15

Table 1: Dataset statistics

in the next section.

3.4 Reranker Model

Our reranker model consists of a pre-trained sentence transformer model, followed by a stack of n standard transformers encoder layers (multi-head self attention plus a feedforward layer, interlaced with skip connections and layer norms), and a final regression layer on the top. While this design is very similar to a hierarchical transformer used in work such as Pappagari et al. (2019) to directly predict the labels, here it is trained to perform a completely different task.

Instead, our reranker model is trained to predict the distribution Relevancy_x. As input to the reranking is the entire collection of sentences $[x_1; \ldots; x_n]$, the stack of encoder layers computes sentence-level self-attention, allowing the model to predict the relevancy of each sentence using surrounding information. We then compare the output of the regression layer with Relevancy_x using KL-divergence loss:

$$\mathcal{L}_{\text{rerank}}(x) = \text{KLDiv}\left(\text{Relevancy}_x, \text{Reranker}(x)\right).$$

After the reranker is trained, we run the entire dataset through the reranker, and obtain its shortened version by selecting the top-k sentences with the highest relevancy scores. Note that the sentence order is especially preserved to improve overall coherence.

3.5 Classification Model

Using the obtained short document dataset, a standard transformer-based classification model is fine-tuned as usual, using either cross-entropy or binary cross-entropy depending on the label type.

3.6 Inference

During inference, the reranker first selects the top-k sentences of a long document, which are then concatenated as the input to the classifier. Importantly, note that the cross-encoder and LLM used to generate the relevancy scores are not needed, thus allowing the deployed model to be non-reliant of LLMs during inference.

4 Experiments

4.1 Data

We test our proposed framework on the following four commonly used datasets for long document classification:

- **Hyperpartisan** (Kiesel et al., 2019) is a binary classification dataset with 645 documents, classifying whether the text is hyperpartisan or not;
- **20 News Groups** (Lang, 1995) is a multiclass dataset with 18,846 documents and 20 balanced classes, representing the domain of a news document (such as religion or sports);
- **EURLEX** (Chalkidis et al., 2019) is a multilabel dataset with 57,000 documents and 4,271 classes, where the documents were originated from EU legal documents and the classes ranges from location (e.g. “France") to specialized domain (e.g. “tariff quota");
- **Book Summaries** (Bamman and Smith, 2013) is a multilabel dataset with 12,788 documents and 227 classes, where the documents are summaries of books extracted from Wikipedia and the classes are broadly speaking the genres of the book (e.g. “Science Fiction" or “Novel").

The dataset statistics are illustrated in Table 1. All data can be found on the official implementation of Park et al. (2022)[8]. We use the same preprocessing scripts provided, including train-test splits whenever applicable, for fair comparisons.

4.2 Baselines

We compare our results with the following baselines:

[8] https://github.com/amazon-science/efficient-longdoc-classification/tree/main

Method	Hyperpartisan	20 News Groups	EURLEX	Book Summaries
BERT (w. Truncation)	92.00	84.79	73.09	58.18
BERT (w. TextRank)	91.15	84.99	72.87	58.94
BERT (w. Random)	89.23	84.65	73.22	59.36
Longformer	95.69	83.39	54.53	56.53
ToBERT	89.54	85.52	67.57	58.16
CogLTX	94.77	84.63	70.13	58.27
H3	94.20	85.90	**76.70**	60.90
Ours (w. BERT)	95.55	86.47	75.39	60.36
Ours (w. RoBERTa)	**97.43**	**87.24**	76.51	**61.77**

Table 2: Accuracy and Micro F1-Scores for four different datasets. Highest value is bolded. Baseline results are taken from Park et al. (2022) and Lu et al. (2023).

1. **BERT (with Truncation)** simply takes the first 512 tokens (including the standard `[CLS]` token prepended in front) of the text input and ignores the remaining tokens;

2. **BERT (with TextRank (Mihalcea and Tarau, 2004))** obtains the BERT representation of the first 512 input tokens, and concatenate it with the representation of up to another 512 tokens from the top ranked sentences using TextRank, an unsupervised sentence ranking algorithm. A linear classifier is added on top. Sentence tokenization and TextRank is done via SpaCy[9];

3. **BERT (with Random Choice)** is similar to BERT with TextRank, but the augmented sentences are simply selected at random. Interesting this often seems to perform as well as TextRank;

4. **Longformer** (Beltagy et al., 2020) is a transformer architecture that replaces full attention with a combination of local and global attention, thus achieving near linear complexity for the attention computation. It has a context window of 4,096, which is enough for all the aforementioned datasets;

5. **ToBERT** (Pappagari et al., 2019) has a hierarchical transformer architecture that considers any arbitrary document as chunks of 200 tokens. Each chunk is passed through a vanilla BERT model, before all output representations are fed through another transformer layer as the final document representation;

6. **CogLTX** (Ding et al., 2020) aims to select key blocks or sentences from the input document. It trains two BERT models, one for classification and one to judge the relevancy of the selected blocks. The judge can trained in a supervised or unsupervised manner depending on the data and task.

7. **H3** (Lu et al., 2023) uses state-space models as an alternative to model sequential data to direct bypass the token limit issue faced in transformer-based models.

4.3 Setup

We detail our models and hyperparameters choices as follows.

Sentence tokenization is done with Stanza[10]. For the choice of cross-encoder, we use `ms-marco-MiniLM-L-12-v2`[11], pre-trained on the MS MARCO Passage Retrieval dataset. The model `all-mpnet-base-v2`[12] is used as our sentence transformer model. Finally we test using both `bert-base-uncased`[13] and `RoBERTa-base`[14] as our classifier.

For the choice of LLM, we use the latest *gpt-35-turbo* version provided by OpenAI, accessed via Microsoft Azure. Note that a small portion of the data triggers the safety content (usually due to violence or sexual reasons), and is particularly common in the Hyperpartisan and Book Summaries datasets. They are simply discarded during the training of the reranker but kept for training the classifier.

[9] https://spacy.io/

[10] https://stanfordnlp.github.io/stanza/

[11] https://sbert.net/docs/cross_encoder/pretrained_models.html

[12] https://sbert.net/docs/sentence_transformer/pretrained_models.html

[13] https://huggingface.co/google-bert/bert-base-uncased

[14] https://huggingface.co/FacebookAI/roberta-base

Dataset	# Sentences	Top-32	Top-16	Top-8	Top-4
20 News Groups	21.01 ± 36.18	86.55	86.47	86.04	85.69
EURLEX	7.77 ± 8.344	74.03	75.28	75.39	64.48

Table 3: Accuracy and Micro F1-Scores for different values of top-k.

We set the number of encoder layers in the reranker n to 2 for all cases, and the number of sentences top-k to 8 for EURLEX and 16 for the remaining three. The reranker model is trained with learning rate 3e-5 until convergence, and the highest validation score – computed via the average relevancy between the top-k sentences and the CoT explanation – is kept. For the classifier, we run a simple grid search of $[1e-5, 2e-5, 5e-5]$ for learning rate, training until convergence and keeping the model with the highest validation score for testing. AdamW is used as the optimizer; batch size is effectively set to 16 throughout, using gradient accumulation if necessary.

5 Results & Analysis

5.1 Overview

Experimental results are shown in Table 2. Following all previous work, multiclass datasets are evaluated using accuracy, whereas multilabel datasets are evaluated using micro F1-score. Our results are the average of three runs. Compared with BERT-based results, our method (c.f. Ours (w. BERT)) is superior on three datasets, and on par on the remaining one; compared with baseline using state-space models, our results (c.f. Ours (w. RoBERTa)) are also superior on three datasets, and on par on the remaining one.

5.2 Comparison with Transformer Baselines

Compared with all previous transformer baselines, our method (Ours (w. BERT)) consistently outperforms in three of the four datasets, while remaining highly competitive on the Hyperpartisan dataset. Taking the best baseline for each dataset (i.e. Longformer for Hyperpartisan, ToBERT for 20 News Groups etc), our method scores 1.00% higher on average, with a maximum of 2.17% gain in EURLEX.

In particular, compared with CogLTX – conceptually the most similar method – our method is significantly superior, averaging 2.49% across all four datasets. This shows the efficacy of using RAG-inspired methodologies to select the most significant sections of a text for long document classification.

5.3 Comparison with Non-Transformer Baselines

To create a fair comparison with the best non-transformer baseline H3, we train our models using RoBERTa-base as the underlying language model, which has the same parameters as BERT but generally performs better. Under these choices, again our method is superior on three of the four datasets with EURLEX as the sole exception, where our results are also highly competitive. On average, our method sees an 1.31% gain in performance.

5.4 Comparison between Datasets

It is interesting to note that there is a significant difference in our performance for multiclass and multilabel datasets. For the two multiclass datasets, we observe an 1.54% increase in performance over the best baseline, whereas for the remaining two multilabel datasets, we only observe a 0.34% increase. While the difference of metrics (accuracy versus micro F1-Score) is one possible explanation, the underlying nature of our approach is also worth considering. In multiclass environments, selecting the most significant 16 sentences is often more than sufficient to predict one correct label, and using as few as four sentences can still gives satisfactory results (see Ablation Studies below). However, when the ground truth contains multiple labels (up to 26 in EURLEX), it naturally becomes much more difficult for a classifier to predict all of them correctly given only eight or 16 sentences. This explanation is also supported by a similarly weaker performance of CogLTX, another chunk selection baseline, in the two multilabel datasets.

5.5 Ablation Studies

Table 3 shows the effect of varying k, the number of sentences selected by the reranker, in one multiclass and one multilabel dataset. For the 20 News Groups, we see that lowering the value of k has a much less impact on the accuracy, despite having a much higher number of sentences on average.

Simply selecting the top-4 sentences retains 85.69 accuracy.

On the other hand, although EURLEX only has 8 sentences per document on average, we observe a steep dropoff in F1-score when using the top-4 sentences for classification. This further explains the aforementioned performance difference in multiclass and multilabel datasets, and thus is important to consider in practical application. It may also be possible to dynamically select k based on the dataset, or even the input document, but we leave that as future work.

6 Limitations

One major limitation is the reliance on LLMs during the training phase. In particular, acquiring a CoT explanation of the text classification, and hence the relevancy labels, involves prompting an LLM. This implies that the dataset needs high quality description of the labels themselves (as opposed to a simple numerical representation of each class, which would be acceptable inputs to all baselines tested in our work). For example, although ECtHR is another widely used dataset for long document classification, we could not test on it due to the lack of correspondence between the numerical labels and the meaning. Furthermore, LLMs are known to perform best in some selective high-resource languages (especially English), which may further limit the application of our method to low-resource languages.

7 Conclusion

Long document classification is important in many domains such as legal or biomedical, where texts, compounded with domain-specific terminology, are often far longer than the standard 512 input tokens of most pre-trained language models. To process such documents, we propose a long document classification method, inspired by recent techniques used by LLMs for RAG. By training a reranker, we select the most significant sections of a document and train a classifier on the selected text. We observe improvements on average for four standard datasets, compared with both transformer and non-transformer baselines.

Future work can aim to dynamically to select top-k, as discussed above, to tailor for the characteristics of different datasets. It is also worth considering the possibility of combining the reranker and classifer into one model, trained end-to-end. This could allow the reranker to assign relevancy scores not only based on the teacher LLM, but also the usefulness in downstream classification.

References

D. Bamman and N. Smith. 2013. New alignment methods for discriminative book summarization. *https://arxiv.org/abs/1305.1319*.

I. Beltagy, M. Peters, and A. Cohan. 2020. Longformer: The long-document transformer. *https://arXiv:2004.05150*.

I. Chalkidis, E. Fergadiotis, P. Malakasiotis, and I. Androutsopoulos. 2019. Large-scale multi-label text classification on EU legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6314–6322, Florence, Italy. Association for Computational Linguistics.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT 2019*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

M. Ding, C. Zhou, H. Yang, and J. Tang. 2020. CogLTX: Applying BERT to long texts. In *Advances in Neural Information Processing Systems 33*, pages 12792–12804.

Q. Dong, L. Li, D. Dai, C. Zheng, Z. Wu, B. Chang, X. Sun, J. Xu, L. Li, and Z. Sui. 2023. A survey on in-context learning. *https://arxiv.org/abs/2301.00234*.

L. Gao, X. Ma, J. Lin, and J. Callan. 2023a. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777, Toronto, Canada. Association for Computational Linguistics.

Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, and H. Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *https://arxiv.org/abs/2312.10997*.

L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *https://arxiv.org/abs/2311.05232*.

J. Kiesel, M. Mestre, R. Shukla, E. Vincent, P. Adineh, D. Corney, B. Stein, and M. Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

K. Lang. 1995. Newsweeder: learning to filter netnews. In *Proceedings of the Twelfth International Conference on ICML*, pages 331–339, San Francisco, CA, USA.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880. Association for Computational Linguistics.

N. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. 2023. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

P. Lu, S. Wang, M. Rezagholizadeh, B. Liu, and I. Kobyzev. 2023. Efficient classification of long documents via state-space models. In *Proceedings of the 2023 Conference on EMNLP*, pages 6559–6565, Singapore. Association for Computational Linguistics.

R. Mihalcea and P Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on EMNLP*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

S. Narang, H. Chung, Y. Tay, L. Fedus, T. Fevry, M. Matena, K. Malkan, N. Fiedel, N. Shazeer, Z. Lan, Y. Zhou, W. Li, N. Ding, J. Marcus, A. Roberts, and C. Raffel. 2021. Do transformer modifications transfer across implementations and applications? In *Proceedings of the 2021 Conference on EMNLP*, pages 5758–5773, Punta Cana, Dominican Republic. Association for Computational Linguistics.

R. Pappagari, P. Zelasko, J. Villalba, Y. Carmiel, and N. Dehak. 2019. Hierarchical transformers for long document classification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 838–844. Association for Computational Linguistics.

H. Park, Y. Vyas, and K. Shah. 2022. Efficient classification of long documents using transformers. In *Proceedings of ACL 2022 (Volume 2: Short Papers)*, pages 702–709, Dublin, Ireland. Association for Computational Linguistics.

C. Raffel, N Shazeer, A Roberts, K Lee, S Narang, M Matena, Y Zhou, W Li, and P Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

P. Shaw, Uszkoreit J, and Vaswani A. 2018. Self-attention with relative position representations. In *Proceedings of NAACL-HLT 2018*, pages 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

W. Sun, L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, and Z. Ren. 2023a. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on EMNLP*, pages 14918–14937, Singapore. Association for Computational Linguistics.

X. Sun, X. Li, J. Li, F. Wu, S. Guo, T. Zhang, and G. Wang. 2023b. Text classification via large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8990–9005, Singapore.

H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

S. Wang, B. Li, M. Khabsa, H. Fang, and H. Ma. 2020. Linformer: Self-attention with linear complexity. *https://arxiv.org/pdf/2006.04768*.

L. Zhang, Y. Zhang, D. Long, P. Xie, M. Zhang, and M. Zhang. 2023. TSRankLLM: A two-stage adaptation of LLMs for text ranking. *https://arxiv.org/abs/2311.16720*.

8 Appendix

8.1 Prompts Used

The following prompts are used for each dataset to obtain relevancy labels:

1. **Hyperpartisan** & **20 News Groups**: "The classification of the following paragraph is known to be '{label}'. \n\n{text}\n\n Write an response explaining why it is classified as {label}. Give references to phrases or sentences from the original text whenever possible".

2. **EURLEX** & **Book Summaries**: "The classifications of the following paragraph is known to be '{labels}'. \n\n{text}\n\n For each label in '{labels}', write an response explaining why it is classified as such. Give references to phrases or sentences from the original text whenever possible".