On The Adaptation of Unlimiformer for Decoder-Only Transformers

Kian Ahrabian[1]*, Alon Benhaim[2], Barun Patra[2]
Jay Pujara[1], Saksham Singhal[2], Xia Song[2]
[1] Information Sciences Institute, University of Southern California
[2] Microsoft
ahrabian@usc.edu, {alonbenhaim,barun.patra}@microsoft.com
jpujara@isi.edu, {saksham.singhal,xiaso}@microsoft.com

Abstract

One of the prominent issues stifling the current generation of large language models is their limited context length. Recent proprietary models such as GPT-4 and Claude 2 have introduced longer context lengths, 8k/32k and 100k, respectively; however, despite the efforts in the community, most common models, such as LLama-2, have a context length of 4k or less. Unlimiformer (Bertsch et al., 2023) is a recently popular vector-retrieval augmentation method that offloads cross-attention computations to a kNN index. However, its main limitation is incompatibility with decoder-only transformers out of the box. In this work, we explore practical considerations of adapting Unlimiformer to decoder-only transformers and introduce a series of modifications to overcome this limitation. Moreover, we expand the original experimental setup on summarization to include a new task (i.e., free-form Q&A) and an instruction-tuned model (i.e., a custom 6.7B GPT model). Our results showcase the effectiveness of these modifications on summarization, performing on par with a model with 2x the context length. Moreover, we discuss limitations and future directions for free-form Q&A and instruction-tuned models.

Keywords: Large Language Models, Decoder-Only Transformers, Retrieval-Augmented Attention

1. Introduction

In recent years, large language models (LLMs) have become critical to many language-based technologies, such as conversational and search systems. LLMs have shown state-of-the-art performance on sequence-to-sequence downstream tasks such as summarization and question-answering. However, the performance of these models is bounded by the information that can fit in their context (see Figure 3 and Section 4.3). Despite the efforts in the community (Choromanski et al., 2021; Beltagy et al., 2020; Ivgi et al., 2023), most of the common open-source models, e.g., MPT (Team, 2023), Falcon (Penedo et al., 2023), and LLama-2 (Touvron et al., 2023), have a context length of 4096 or less. As such, efficiently overcoming this limitation would allow a broader and fairer adaptation of LLMs while increasing their performance across benchmarks.

In general, most of the existing methods for extending the contextual information in LLMs focus on one of the following approaches: 1) extending positional embeddings through extrapolation or interpolation (Press et al., 2022; Sun et al., 2023), 2) introducing recurrence in the transformer (Hutchins et al., 2022; Yang et al., 2019), 3) introducing sparsity in the attention mechanism (Beltagy et al., 2020; Zaheer et al., 2020), and 4) augmenting the transformer with a vector-retrieval module (Rubin and Berant, 2023). One popular approach that has gained much traction recently for extending contextual information to unlimited inputs (theoretically) is Unlimiformer (Bertsch et al., 2023). Unlimiformer is a vector-retrieval augmentation method that offloads the cross-attention computations to a kNN

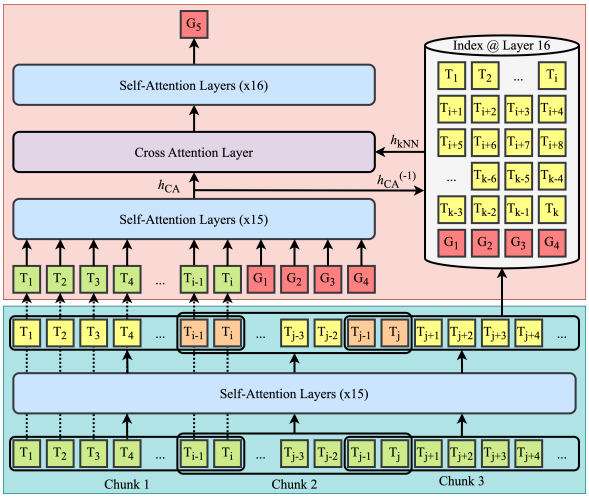

Figure 1: Overview of an example of the adapted decoder-only model where only a single layer (e.g., 16th) uses cross-attention. Here, T_i and G_i represent the original context and generated tokens. The cyan section encapsulates the first pass to create the indices, while the pink section illustrates the second pass to generate sequences. Note that the first chunk appears in both the input and the index by design, keeping the input the same in all variations of our experiments.

* Work done during an internship at Microsoft.

LREC-COLING 2024, pages 12395–12402

index and can wrap *any existing encoder-decoder model*. However, its incompatibility with decoder-only models is a significant shortcoming.

Contributions: In this work, we present a set of modifications to overcome this limitation of the Unlimiformer architecture, adapting it to the decoder-only models (see Figure 1). These modifications consist of 1) modifying the cross-attention formulation to include information fusion, 2) updating the index creation procedure, 3) addressing the index staleness problem, and 4) adapting the chunk encoding procedure to causal attention (see Section 3). Moreover, we introduce a new evaluation setting and present our experiments on four long-document datasets across two tasks: summarization and free-form Q&A. Our experiments show that our modifications improve summarization datasets, performing on par with a model with 2x context length. We also discuss the limitations and future directions for free-form Q&A and instruction-tuned models.

2. Related Works

Many prior works, such as Linformer (Wang et al., 2020) and Reformer (Kitaev et al., 2020), have been focused on creating more efficient transformers. Tay et al. (2022) present a comprehensive study on these models. Moreover, there have been efforts to accelerate dense attention calculations by crafting IO-aware CUDA kernels (Dao et al., 2022; Dao, 2023). Recently, Rubin and Berant (2023) have introduced another retrieval-augmented attention model for decoder-only transformers; however, unlike Unlimiformer, this model does not work in zero-shot settings. Finally, there have been attempts to break away from attention-based models entirely through linear RNNs (Peng et al., 2023) or convolutions (Poli et al., 2023).

3. Methodology

In summary, Unlimiformer consists of three main steps: 1) split the input into overlapping digestible chunks, 2) encode each chunk and store the hidden states of the middle-half tokens in a kNN index, and 3) approximate the dense attention in the decoder using a subset of hidden states retrieved from the index. In this section, we discuss our changes to make Unlimiformer compatible with decoder-only models[1].

3.1. Cross-Attention

Formally, in Unlimiformer, the dot-product part of the cross-attention mechanism in decoder layers is approximated as

$$QK^T \approx (h_d W_q W_k^T) h_e^T \tag{1}$$

where h_d is the decoder's hidden states, and h_e is the retrieved hidden states that maximize $(h_d^{(-1)} W_q W_k^T) h_e^T$.

In decoder-only transformers, due to the absence of a natural encoding/decoder splitting layer, we can arbitrarily choose any layer to use cross-attention instead of self-attention. This allows us to use various simple or complex patterns for the set of cross-attention layers, $\mathcal{L}$ (e.g., $\mathcal{L} = \{16\}$ in Figure 1). Appendix B details how these patterns are tuned as hyperparameters of the model.

Let h_{CA} be the input to a cross-attention layer. Based on $h_{\text{CA}}^{(-1)}$, we can retrieve the relevant vectors, h_{kNN}, from the index. Similar to the memory transformer (Burtsev et al., 2020), by fusing h_{kNN} and h_{CA} we form the new query matrix h_q, and the new key-value matrix, h_{kv} as

$$h_q = [h_{\text{kNN}}[\alpha_q :]; h_{\text{CA}}[\beta_q :]] \tag{2}$$

$$h_{kv} = [h_{\text{kNN}}[\alpha_{kv} :]; h_{\text{CA}}[\beta_{kv} :]] \tag{3}$$

where α_q and β_q (α_{kv} and β_{kb}) control the retention sizes of the retrieved and input vectors in the query (key-value) matrix. This fusion scheme gives us more flexibility on what information is processed in the attention mechanism. Given h_q and h_{kv}, we can approximate the dot-product part of the cross-attention as

$$QK^T \approx (h_q W_q)(h_{kv} W_k)^T . \tag{4}$$

Similar to the original approach, W_q and W_k are head-specific but use the same index.

3.2. kNN Indices

Without separate encoder/decoder modules, operating with only one index for the whole model is impossible. This is because each layer in a decoder-only model attends to the output of its previous layer, in contrast to the decoder layers in encoder-decoder models that attend to the encoder's output. Consequently, if we arbitrarily use a specific layer's output for building our index, we create a distributional mismatch between the expected input and actual inputs of future layers. Hence, we must create a separate index for each cross-attention layer to overcome this issue. However, this approach requires much more memory and computation cost to build the indices. For example, in our models, it could add up to $|\mathcal{L}|$ times the cost of having a single index. Section 8 discusses the potential ways of mitigating this issue. In our experiments, we only tune our models to use at most three cross-attention layers, i.e., $|\mathcal{L}| = 3$.

[1] Concurrently, Bertsch et al. (2023) have released an implementation for decoder-only models. Appendix A details the differences in our adaptation.

Article:
Title
Minimally Supervised Learning of Affective Events Using Discourse Relations

Abstract
Recognizing affective events that trigger positive or negative sentiment has a wide range of natural language processing applications but remains a...

Question:
What is the seed lexicon?

Answer:
A vocabulary of positive and negative predicates that helps determine the polarity score of an event.

Figure 2: A sample free-form Q&A prompt. The article section consists of the truncated version of the full article that fits in the context. The question section is always fully included.

3.3. Index Staleness

One of the potential issues of a static index is staleness. Specifically, since Unlimiformer approximates dense attention, without updating the index, we would lose the information from the newly generated tokens, which might lead to incoherent outputs. For example, assume we have a model with a context length of N. In this scenario, at generation step $N+2$, the input to the model would be the last N generated tokens, effectively discarding the first generated token as it is also absent from the index. To fix this problem, at each generation step and each layer, we add $h_{\text{CA}}^{(-1)}$ to its respective index. This ensures the addition of the most recent token and keeps the indices from going stale. Appendix 6.1 presents an ablation study that showcases the effectiveness of this simple change.

3.4. Chunks Encoding

In contrast to encoder-decoder models, decoder-only transformers use causal (unidirectional) attention. This difference means that a token has seen enough contextual information if a certain number of tokens are behind it. As a result, instead of only storing the hidden states of the middle half tokens, we can keep all the non-overlapping ones. This will allow us to be slightly more efficient when processing long documents. Note that only the first instance of overlapping tokens is added to the index, as illustrated by orange tokens in Figure 1.

4. Experimental Setup

4.1. Datasets & Tasks

For our experiments, we use datasets from the two tasks of summarization and free-form Q&A, chosen due to the existence of long-document benchmarks (Shaham et al., 2023). Specifically, for summarization, we use GovReport (GAO) (Huang et al., 2021) and BookSum (Kryscinski et al., 2022), and for free-form Q&A, we use NarrativeQA (Kočiský et al., 2018) and Qasper (Dasigi et al., 2021). Moreover, we tune the hyperparameters on the validation sets and report the results on the test sets. All the experiments are carried out in a zero-shot setting. Table 1 presents the statistics of these datasets.

Dataset	Metrics	#Samples	Avg #Tokens
GovReport (GAO)	ROUGE-{1,2,L},	611	11395
BookSum	METEOR	46	164695
NarrativeQA	Token F1	10557	76433
Qasper		1372	4836

Table 1: Statistics of the datasets. The average number of tokens is obtained using the GPT-4 tokenizer: https://github.com/openai/tiktoken.

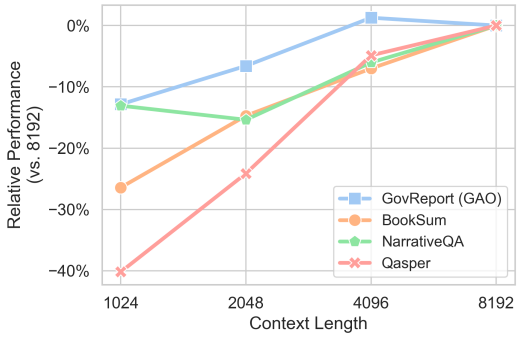

Figure 3: Performance comparison of GPT-Summ on GovReport (GAO) and BookSum, and GPT-Inst on NarrativeQA and Qasper at varying context lengths.

4.2. Models

We used two distinct base models to evaluate our modifications: 1) **GPT-Summ:** a finetuned summarization model and 2) **GPT-Inst:** an instruction-tuned model. To better understand the impact of our approach compared to dense attention, in contrast to prior works, we pre-train both variants of the models with a sequence length of 8192 using the same architecture as the GPT-3 6.7B model (Brown et al., 2020b), with the addition of RoPE embeddings (Su et al., 2021).

4.3. Evaluation Setup

The original Unlimiformer paper presents two main experimental comparisons: 1) $\text{BART}_{\text{base}}$ vs. $\text{BART}_{\text{base}}$ + Unlimiformer and 2) $\text{BART}_{\text{base}}$ + Unlimiformer vs. SLED (Ivgi et al., 2023) and Longformer (Beltagy et al., 2020). These comparisons showcase the effectiveness of the

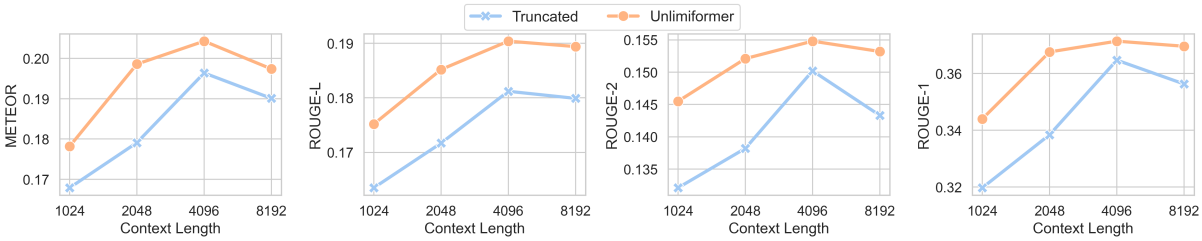

Figure 4: METEOR and ROUGE-{1,2,L} achieved by GPT-Summ on the GovReport (GAO) dataset.

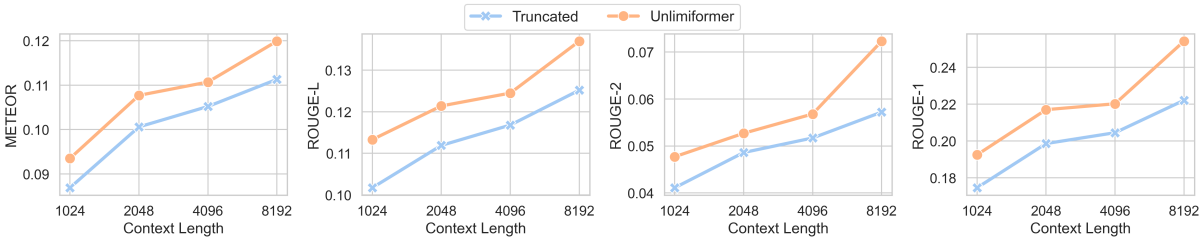

Figure 5: METEOR and ROUGE-{1,2,L} achieved by GPT-Summ on the BookSum dataset.

proposed model; however, one missing crucial evaluation setup is the comparison to the same base model with longer context lengths, e.g., GPT-Summ$[2048]$ vs. GPT-Summ$[1024]$ + Unlimiformer. Such a setup provides more insight into how efficiently Unlimiformer uses the extra information provided through the kNN index. In this work, we focus on this setting by restricting the context length of the base model to 1024, 2048, 4096, and 8192 tokens and then comparing them to variations equipped with Unlimiformer. To ensure that our models and datasets showcase meaningful differences across context lengths in such a setup, we ran experiments on the base models using the validation sets, estimating the performance changes as contextual information increased. Figure 3 illustrates the results of our experiments. Moreover, Appendix 6.2 presents a case study to ensure the performance disparity in Figure 3 is an artifact of the datasets, not a deficiency in the models.

4.4. Prompt Structure

For the free-form Q&A datasets and the instruction-finetuned model, we opted for a simple three-part (Article, Question, and Answer) template. Figure 2 illustrates an example of the prompt structure. We also investigated ZeroScrolls's prompt template (Shaham et al., 2023), but since we did not notice any significant difference in the performance, we continued with the more straightforward template.

4.5. Metrics

For summarization, we report ROUGE-{1,2,L} (Lin, 2004) and METEOR (Banerjee and Lavie, 2005) as 1) they are standard metrics and 2) have shown good correlation to expert annotations (Fabbri et al., 2021). Moreover, in our early experiments, we investigated BERTScore (Zhang* et al., 2020); however, similar to Bertsch et al. (2023)'s findings, we observed a lack of distinguishing power among the lengthy summaries, even when other metrics and manual inspection showed improvements. For free-form Q&A, we used the standard token F1 metric, similar to the ZeroScrolls benchmark.

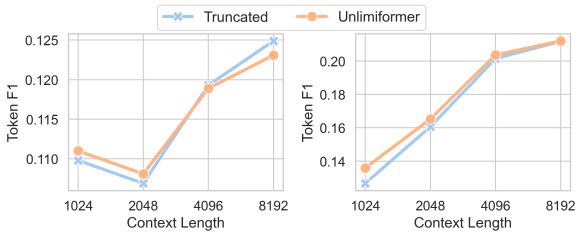

Figure 6: Token F1 achieved by GPT-Inst on NarrativeQA (left) and Qasper (right) datasets.

5. Results

5.1. Summarization

Figures 4 and 5 present our experimental results using the GPT-Summ model on the GovReport (GAO) and BookSum datasets, respectively. As evident from these results, adding our modifications improves the model's performance to a 2x level (e.g., GPT-Summ$[1024]$ + Unlimiformer $\approx$ GPT-Summ$[2048]$) on the GovReport (GAO) dataset. Similarly, we observe significant improvements in the BookSum dataset.

Variation	Rouge-1	Rouge-2	Rouge-L
w/o	0.4247	0.1734	0.2512
w/	**0.4263**	**0.1744**	**0.2523**

Table 2: Effect of adding newly generated tokens to the index on the performance of the model.

Chunk	Min Evidence > 1024 (%)	Min Evidence > 2048 (%)	Min Evidence > 4096 (%)	Min Evidence > 8192 (%)
Train	66.37	47.79	19.26	1.42
Valid	67.77	47.03	15.30	0.87
Test	68.22	46.02	13.71	0.75

Table 3: Percentage of answers with minimum evidence position outside a given context length in Qasper.

These results showcase the effectiveness of our proposed modifications.

5.2. Free-Form Q&A

Figure 6 illustrates our experimental results using the GPT-Inst model on the NarrativeQA and Qasper datasets. Although we can observe some improvements in both Qasper and (mostly) NarrativeQA, they are less significant than the summarization datasets' results. Given the prompt-based approach used for the free-from Q&A task (see Figure 2), the insignificant improvements could be an artifact of the instruction-tuned model being too biased toward the input, making it insensitive to the added information in the cross-attention layers. These results present exciting opportunities to investigate such models in future studies.

6. Ablations

6.1. Index Staleness

To study the effect of index staleness, we experiment with an internal summarization dataset consisting of samples of up to 7k tokens using a model with a context length of 2048 and a generation limit of 700 tokens. Table 2 presents the result of our experiments. Although the generation length is still way under the 2048 limit, we can see a slight positive improvement in the performance, showcasing the usefulness of this simple addition with almost no cost. Moreover, we believe the performance boost will increase as the generation length increases.

6.2. Contextual Information

In the Qasper dataset, we have access to a set of evidence for each answer. Hence, we can calculate the percentage of answers that all of their evidence falls out of the range of a specific context length. Table 3 presents these numbers for different context lengths. These numbers are consistent with the improvements in Figure 3, which showcase the validity of our models.

7. Conclusion

This work presented a set of changes to adapt the Unlimiformer architecture to decoder-only models. We evaluated these changes with a new set of experiments, showcasing their effectiveness, especially on summarization datasets. We also identified a failure case that warrants further investigations in future studies. We hope this work paves the way for the broader use of this architecture.

8. Limitations

Query Bias Since both Unlimiformer and our approach use a specific query vector to retrieve hidden states from the index, the retrieval process becomes biased on the query vector. In Unlimiformer, this vector is $h_d^{(-1)}$, which is calculated by attending to the generated tokens. In our approach, this vector is $h_{\text{CA}}^{(-1)}$, which is calculated by attending to the whole input, i.e., original context + generated tokens. This dependence on the original context potentially reduces the expected performance gains when external indices are used. Moreover, it could partially explain the lack of significant improvements in Section 5.2.

Latency The main setback of having many indices is the increase in latency. To alleviate this problem, we could 1) use approximate indices and/or 2) use indices that operate on GPU to remove the CPU-GPU transfer overhead.

9. Ethical Considerations

This paper does not have any ethical considerations

10. Bibliographical References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew R Gormley. 2023. Unlimiformer: Long-range transformers with unlimited length input. *arXiv preprint arXiv:2305.01625*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020a. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020b. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Mikhail S Burtsev, Yuri Kuratov, Anton Peganov, and Grigory V Sapunov. 2020. Memory transformer. *arXiv preprint arXiv:2006.11527*.

Krzysztof Marcin Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Quincy Davis, Afroz Mohiuddin, Lukasz Kaiser, David Benjamin Belanger, Lucy J Colwell, and Adrian Weller. 2021. Rethinking attention with performers. In *International Conference on Learning Representations*.

Tri Dao. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Tri Dao, Daniel Y Fu, Stefano Ermon, Atri Rudra, and Christopher Re. 2022. Flashattention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexander R. Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. SummEval: Re-evaluating Summarization Evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. 2022. Block-recurrent transformers. In *Advances in Neural Information Processing Systems*.

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

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Wojciech Kryscinski, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. 2022. BOOKSUM: A collection of datasets for long-form narrative summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6536–6558, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke 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, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{bert}a: A robustly optimized {bert} pretraining approach.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, et al. 2023. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint arXiv:2305.13048*.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. 2023. Hyena hierarchy: Towards larger convolutional language models. *arXiv preprint arXiv:2302.10866*.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

Ohad Rubin and Jonathan Berant. 2023. Long-range language modeling with self-retrieval. *arXiv preprint arXiv:2306.13421*.

Uri Shaham, Maor Ivgi, Avia Efrat, Jonathan Berant, and Omer Levy. 2023. Zeroscrolls: A zero-shot benchmark for long text understanding. *arXiv preprint arXiv:2305.14196*.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2023. A length-extrapolatable transformer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14590–14604, Toronto, Canada. Association for Computational Linguistics.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2022. Efficient transformers: A survey. *ACM Comput. Surv.*, 55(6).

MosaicML NLP Team. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms. Accessed: 2023-10-05.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

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*, 33:17283–17297.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

A. Implementation Comparison

Although similar in using separate indices for each layer, we present additional modifications to the original methodology[2]. Specifically, 1) we introduce an update procedure for indices to avoid staleness, 2) we use a slightly more efficient chunk encoding approach, and 3) we introduce information fusion into the architecture and reformulate the attention calculations to be more comprehensive (see Section 3.1). Moreover, to the best of our knowledge, no evaluation of their proposed

[2] https://github.com/abertsch72/unlimiformer

methodology has been presented for decoder-only models, a shortcoming that our work aims to address using a new evaluation setting and new datasets.

B. Hyperparameters

Cross-Attention Layers ($\mathcal{L}$) To find the best $\mathcal{L}$, first, we find the highest-performing single-layer pattern. Then, we expand that pattern by adding one more cross-attention layer before or after that layer, with varying distances, until no performance improvement can be seen over the validation set. We continue this process up to three layers. For example, if $\mathcal{L} = \{16\}$ is the highest-performing single-layer pattern, at the second step, we will try $\mathcal{L} \in \{\{16, 17\}, \{15, 16\}, \{16, 18\}, \ldots\}$, and then at the third step, assuming $\mathcal{L} = \{16, 18\}$ was the best-performing pattern, we will try $\mathcal{L} \in \{\{14, 16, 18\}, \{16, 18, 20\}, \ldots\}$. In most of our experiments, the best performance was achieved by $\mathcal{L} = \{20, 22, 24\}$.

Retention Sizes ($\alpha_q, \beta_q, \beta_{kv}, \alpha_{kv}$) After tuning $\mathcal{L}$, we tune these hyperparameters by first constraining them with $\alpha_q + \beta_q = \beta_{kv} + \alpha_{kv} = S$, where S is the context length, and then doing a sweep on $\alpha_q, \alpha_{kv} \in \{0.05, 0.1, 0.15, \ldots, 0.95, 1\} \times S$. In most of our experiments, the best performance was achieved by $\alpha_q, \alpha_{kv} = 0.4 \times S$.