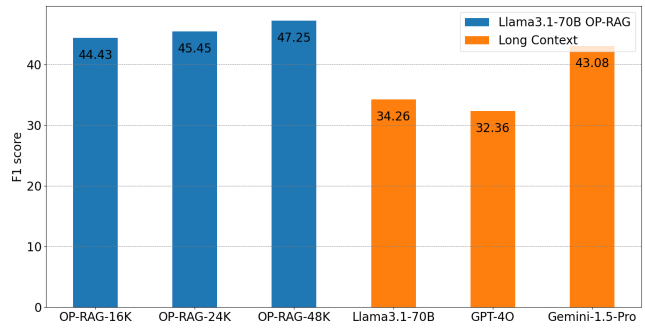
In Defense of RAG in the Era of Long-Context Language Models

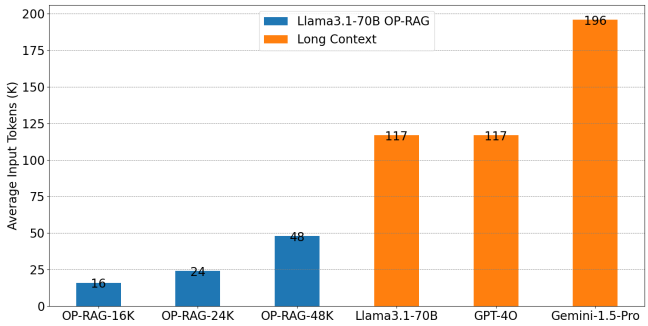
Anonymous submission

Abstract

Overcoming the limited context limitations in early-generation LLMs, retrieval-augmented generation (RAG) has been a reliable solution for context-based answer generation in the past. Recently, the emergence of long-context LLMs allows the models to incorporate much longer text sequences, making RAG less attractive. Recent studies show that long-context LLMs significantly outperform RAG in long-context applications. Unlike the existing works favoring the long-context LLM over RAG, we argue that the extremely long context in LLMs suffers from a diminished focus on relevant information and leads to potential degradation in answer quality. This paper revisits the RAG in long-context answer generation. We propose an order-preserve retrieval-augmented generation (OP-RAG) mechanism, which significantly improves the performance of RAG for long-context question-answer applications. With OP-RAG, as the number of retrieved chunks increases, the answer quality initially rises, and then declines, forming an inverted U-shaped curve. There exist sweet points where OP-RAG could achieve higher answer quality with much less tokens than long-context LLM taking the whole context as input. Extensive experiments on public benchmark demonstrate the superiority of our OP-RAG.



(a) F1 score.



(b) Input token count.

Figure 1: Comparisons between the proposed order-preserve retrieval-augmented generation (OP-RAG) and approaches using long-context LLMs without RAG on En.QA dataset of ∞Bench. Our OP-RAG uses Llama3.1-70B as generator, which significantly outperforms its counterpart using Llama3.1-70B without RAG.

1 Introduction

Due to the limited context window length (*eg*, 4096) of early-generation large language models (LLMs), retrieval augmented generation (RAG) (Guu et al., 2020; Lewis et al., 2020) is an indispensable choice to handle a large-scale context corpus. Since the answer quality is heavily dependent on the performance of the retrieval model, a lot of efforts are devoted to improving the retrieval recall/precision when designing the RAG system.

Recently, the state-of-art LLMs support much longer context windows. For example, GPT-4O (OpenAI, 2023), Claudi-3.5 (Anthropic, 2024), Llama3.1 (Meta, 2024b), Phi-3 (Abdin et al., 2024), and Mistral-Large2 (AI, 2024) all support 128-K context. Gemini-1.5-pro even supports a 1M context window. The recent emergence of long-context LLMs naturally leads to the question: is RAG necessary in the age of long-context LLMs? Li et al. (2024) recently systematically compares RAG with long-context (LC) LLMs (w/o RAG) and demonstrates that LC LLMs consistently outperform RAG in terms of answer quality.

In this work, we re-examine the effectiveness of RAG in long-context answer generation. We observe that the order of retrieved chunks in the

context of LLM is vital for the answer quality. Different from traditional RAG which places the retrieved chunks in a relevance-descending order, we propose to preserve the order of retrieved chunks in the original text. Our experiments show that the proposed order-preserving mechanism significantly improves the answer quality of RAG.

Meanwhile, using the proposed order-preserve RAG, as the number of retrieved chunks increases, the answer quality initially rises and then declines. This is because, with more retrieved chunks, the model has access to more potentially relevant information, which improves the chances of retrieving the correct context needed to generate a high-quality answer. However, as more chunks are retrieved, the likelihood of introducing irrelevant or distracting information also increases. This excess information can confuse the model, leading to a decline in answer quality. The trade-off, therefore, is between improving recall by retrieving more context and maintaining precision by limiting distractions. The optimal point is where the balance between relevant and irrelevant information maximizes the quality of the answer. Beyond this point, the introduction of too much irrelevant information degrades the model's performance. It explains the inferior performance of the approach taking the whole long context as the input of LLM.

Different from the conclusion from Li et al. (2024), with the proposed order-preserving mechanism, RAG achieves higher answer quality compared with its counterparts that rely solely on Long-Context LLMs. As shown in Figure 4a, On En.QA dataset of ∞Bench (Zhang et al., 2024), using only 16K retrieved tokens, we achieve 44.43 F1 score with Llama3.1-70B. In contrast, without RAG, Llama3.1-70B making full use of 128K context only achieves 34.32 F1 score, GPT-4O achieves only 32.36 F1 score and Gemini-1.5-Pro obtains only 43.08 F1 score as evaluated by Li et al. (2024). That is, RAG could achieve a higher F1 score even with a significant reduction on input length.

2 Related Work

Retrieval-augmented generation. By incorporating the external knowledge as context, retrieval-augmented generation (RAG) (Guu et al., 2020; Lewis et al., 2020; Mialon et al., 2023) allows language model to access up-to-date and specific information, reducing hallucinations and improving factual accuracy. Before the era of long-context LLMs, RAG is a promising solution to overcoming the limitation of short context window.

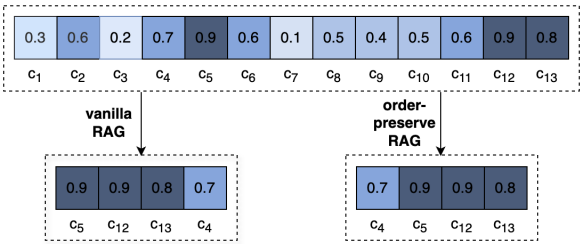


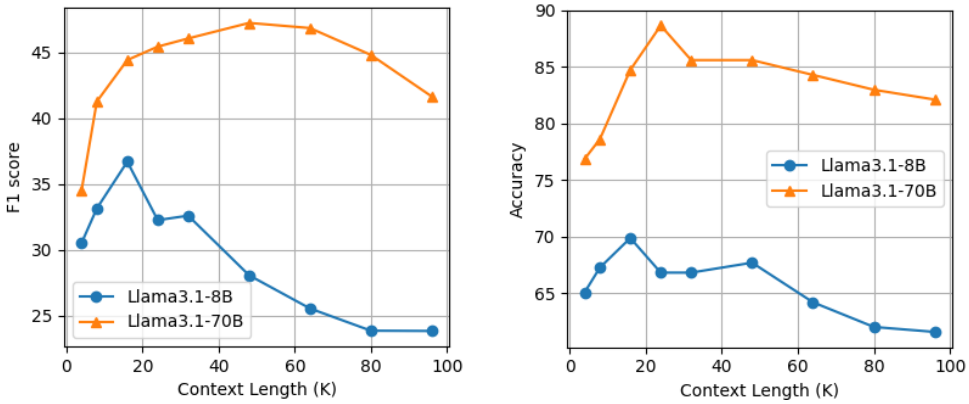
Figure 2: Vanilla RAG versus the proposed order-preserve the RAG. As shown in the example, a long document is cropped into 13 chunks, $\{c_i\}_{i=1}^{13}$. The similarity score is appended to each chunk. We retrieve top 4 chunks with the highest similarity scores. Vanilla RAG places the chunks in a score-descending order, whereas the proposed order-preserve RAG places the chunks based on the order in the original document.

Long-context LLM. To support the long sequence of language models, many efforts have been devoted to improving the computing efficiency of self-attention (Choromanski et al., 2020; Zaheer et al., 2020; Tay et al., 2020; Dao et al., 2022; Dao, 2024) and boosting extensibility of positional encoding (Press et al., 2021; Sun et al., 2022; Chen et al., 2023). Recently, the flagship LLMs such as GPT-4O (OpenAI, 2023), Gemini-1.5-Pro (Reid et al., 2024), Claudi-3.5 (Anthropic, 2024), Grok-2 (xAI, 2024), and Llama3.1 (Meta, 2024a) have supported extremely large context. With the existence of long-context LLMs, RAG is no longer a indispensable module for long-context question-answering task. Recently, Li et al. (2024) concludes that using long-context without RAG could significantly outperforms RAG. Different from the conclusion from (Li et al., 2024), in this work, we demonstrate the proposed order-preserve RAG could beat the long-context LLMs without RAG.

3 Order-Preserve RAG

Let us denote the long textual context, *e.g.*, a long document, by d. We split d into N chunks sequentially and uniformly, $\{c_i\}_{i=1}^{N}$. The index i implies the sequential order of the chunk c_i in d. That is, c_{i-1} denotes the chunk before c_i whereas c_{i+1} denotes the chunk right after c_i. Given a query q, we obtain the relevance score of the chunk c_i by computing cosine similarity between the embedding of q and that of c_i:

$$s_i = \cos(\text{emb}(q), \text{emb}(c_i)), \quad (1)$$



(a) EN.QA (b) EN.MC

Figure 3: The influence of context length on the performance of RAG. The evaluations are conducted on En.QA and EN.MC datasets of ∞Bench.

where $\cos(\cdot,\cdot)$ denotes the cosine similarity function and $\mathrm{emb}(\cdot)$ denotes the embedding function.

We retrieve the top k chunks with the highest similarity scores with the query and denote the indices of top k chunks by $\mathcal{J} = \{j_i\}_{i=1}^{k}$. We preserve the order of chunks in the original long context d, that is, we constrain

$$j_l > j_m \iff l > m. \tag{2}$$

Figure 2 visualizes the difference between the vanilla RAG and the proposed order-preserve RAG. Different from vanilla RAG placing the chunks in the order of similarity descending, the proposed order-preserve RAG keep the order of chunks in the original document.

4 Experiments

4.1 Datasets.

We conduct experiments on EN.QA and EN.MC datasets of ∞Bench (Zhang et al., 2024) benchmark, specially designed for long-context QA evaluation. To be specific, En.QA consists of 351 human-annotated question-answer pairs. On average, the long context in En.QA contains 150,374 words. We use F1-score as metric for evaluation on En.QA. EN.MC consists of 224 question-answer pairs, which are annotated similarly to En.QA, but each question is provided with four answer choices. On average, the long context in En.MC contains 142,622 words. We use accuracy as metric for evaluation on En.MC. We notice there is another benchmark termed LongBench (Bai et al., 2023). Nevertheless, the average context length of LongBench is below 20K words, which is not long enough to evaluate the recent long-context LLMs supporting 128K-token window size.

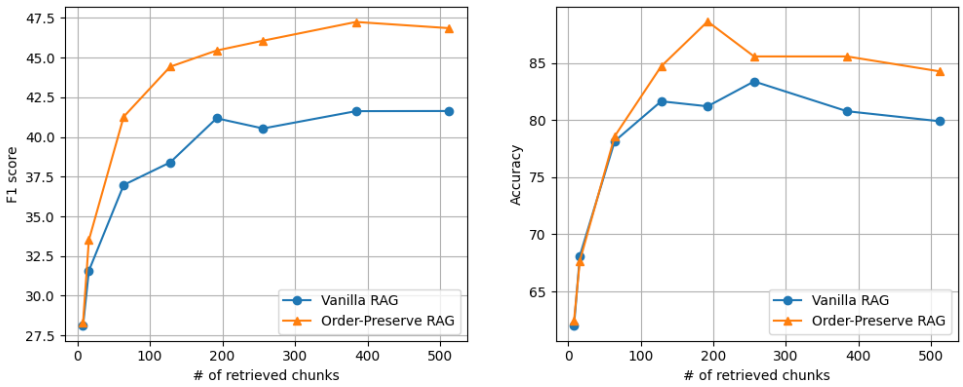
4.2 Implementation details.

We set the chunk size as 128 tokens on all datasets. Chunks are non-overlapped. We use BGE-large-en-v1.5 (Xiao et al., 2023) to extract the embedding of queries and chunks, by default.

4.3 Ablation Study

The influence of context length. We evaluate the influence of the context length on the performance of the proposed order-preserve RAG. Since each chunk contains 128 tokens, the context length is $128m$, where m is the number of the retrieved chunks as the context for generating the answer. As shown in Figure 3, as the context length increases, the performance initially increases. This is because more context might have a greater chance of covering the relevant chunk. Nevertheless, as the context length further increases, the answer quality drops since more irrelevant chunks are used as distractions. To be specific, Llama3.1-8B model achieves the performance peak when the context length is 16K on both EN.QA dataset and EN.MC dataset, whereas the best performance of Llama3.1-70B model is achieved at 48K on EN.QA and 24K on EN.MC. The fact that the peak point of Llama3.1-70B comes later than Llama3.1-8B model might be because the larger-scale model has a stronger capability to distinguish the relevant chunks from irrelevant distractions.

Order-preserve RAG versus vanilla RAG. As



(a) EN.QA (b) EN.MC

Figure 4: Comparisons between the proposed order-preserve RAG and vanilla RAG. The evaluations are conducted on En.QA and EN.MC datasets of ∞Bench, using Llama3.1-70B model.

shown in Figure 4, when the number of retrieved chunks are small (*e.g*, 8), the advantage of the proposed order-preserve RAG over vanilla RAG is not considerably. In contrast, when the number of retrieved chunks is large, our order-preserve RAG significantly outperforms vanilla RAG. To be specific, on EN.QA dataset, when the number of retrieved chunk is 128, vanilla RAG only achieves 38.40 F1-score whereas our order-preserve RAG achieves 44.43 F1-score. On EN.MC dataset, retrieving 192 chunks, vanialla RAG only achieves 81.22 accuracy whereas our order-preserve RAG obtains 88.65 accuracy.

Method	EN.QA		EN.MC	
	F1 Score	Tokens	Acc.	Tokens
Long-context LLM w/o RAG				
Llama3.1-70B	34.26	117K	71.62	117K
GPT-4O	32.36	117K	78.42	117K
Gemini-1.5-Pro	43.08	196K	85.57	188K
SELF-ROUTE (Li et al., 2024)				
GPT-4O	34.95	85K	77.29	62K
Gemini-1.5-Pro	37.51	83K	76.86	62K
Llama3.1-70B order-preserve RAG (ours)				
OP-RAG-16K	44.43	16K	84.72	16K
OP-RAG-24K	45.45	24K	**88.65**	24K
OP-RAG-48K	**47.25**	48K	85.59	48K

Table 1: Comparisons among the long-context LLM without RAG, SELF-ROUTE mechanism (Li et al., 2024) and the proposed order-preserve (OP) RAG.

4.4 Main Results

We compare the proposed order-preserve RAG with two types of baselines. The first category of approaches uses the long-context LLM without RAG. As shown in Table 1, without RAG, LLM takes a huge number of tokens as input, which is inefficient and costly. In contrast, the proposed order-preserve RAG not only significantly reduces the number of tokens, but also significantly improves the answer quality. For instance, using Llama3.1-70B model, the approach without RAG only achieves a 34.26 F1 score on EN.QA with an average of 117K tokens as input. In contrast, our OP-RAG with 48K tokens as input attains a 47.25 F1 score. The second category of baselines takes the SELF-ROUTE mechanism (Li et al., 2024), which routes queries to RAG or long-context LLM based on the model self-reflection. As shown in Table 1, ours significantly outperforms them using much fewer tokens in the input of LLMs.

5 Conclusion

In this paper, we have revisited the role of retrieval-augmented generation (RAG) in the era of long-context language models (LLMs). While recent trends have favored long-context LLMs over RAG for their ability to incorporate extensive text sequences, our research challenges this perspective. We argue that extremely long contexts in LLMs can lead to a diminished focus on relevant information, potentially degrading answer quality in question-answering tasks. To address this issue, we proposed the order-preserve retrieval-augmented generation (OP-RAG) mechanism. Our extensive experiments on public benchmarks have demonstrated that OP-RAG significantly improves the performance of RAG for long-context question-answer applications. OP-RAG's superior performance suggests that efficient retrieval and focused context utilization can outperform the brute-force approach of processing extremely long contexts.

6 Limitation

The proposed method only considers the English evaluation benchmark. Its effectiveness on the other languages are not evaluated. Meanwhile, the experiments are conducted based on the open-source Llama3.1 model, and the effectiveness of the proposed method on the close-source models like GPT4 and Gemini are not evaluated due to the limited budget.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Mistral AI. 2024. Mistral large 2.

Anthropic. 2024. Claude 3.5 sonnet.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*.

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*.

Tri Dao. 2024. FlashAttention-2: Faster attention with better parallelism and work partitioning. In *International Conference on Learning Representations (ICLR)*.

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

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. 2024. Retrieval augmented generation or long-context llms? a comprehensive study and hybrid approach. *arXiv preprint arXiv:2407.16833*.

Meta. 2024a. Introducing llama 3.1: Our most capable models to date.

Meta. 2024b. Llama 3.1 models.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. 2023. Augmented language models: a survey. *arXiv preprint arXiv:2302.07842*.

OpenAI. 2023. GPT-4 technical report. *ArXiv*, 2303:08774.

Ofir Press, Noah A Smith, and Mike Lewis. 2021. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2022. A length-extrapolatable transformer. *arXiv preprint arXiv:2212.10554*.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey.(2020). *arXiv preprint cs.LG/2009.06732*.

xAI. 2024. Grok-2 beta release.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

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.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Khai Hao, Xu Han, Zhen Leng Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. 2024. ∞bench: Extending long context evaluation beyond 100k tokens. *Preprint*, arXiv:2402.13718.