AESS: A Simple Method to Model Long Context in Large Language Models

Anonymous ACL submission

Abstract

As Large Language models (LLMs) gain popularity, the need to understand long texts continues to grow. Despite many models now extending the context window several times beyond the base model, the performance of these models in processing long texts varies across different tasks. Therefore, we propose Attention Entropy Sort and Selection (AESS) to address the long text problem. Our method achieves length generalization of LLM by leveraging the large model itself to retrieve the most relevant information for the task when the context window is limited. Moreover, this method is task-agnostic, and different tasks only need different prompts to achieve their retrieval. Results from the LongBench benchmark show that AESS can improve LLM performance by 9-10% compared to other retrieval methods. Furthermore, our method can also be adapted to various models and improve performance. Therefore, AESS is a promising solution for various applications that require LLMs to handle tasks with lengthy inputs effectively.

1 Introduction

Large Language models (LLMs) (Radford et al., 2018; Zhang et al., 2022; Touvron et al., 2023) serve as vital components in various natural language processing applications such as dialog interfaces (Taori et al., 2023; Chiang et al., 2023), automatic translator (Peng et al., 2023b; Lu et al., 2023), summarization tools (Goyal and Durrett, 2020), and question answering (Kamalloo et al., 2023). They primarily perform tasks through prompts, where task instructions and data are presented as text, and the model generates a text-based response. Incorporating extensive input contexts with thousands of tokens is common when utilizing language models for lengthy inputs like chat history, as well as for enhancing them with external information such as relevant documents from a search engine or database query results (Petroni et al., 2020; Ram et al., 2023; Shi et al., 2023; Mallen et al., 2023; Schick et al., 2023). It's challenge for LLMs to efficiently and accurately tackle long sequences.

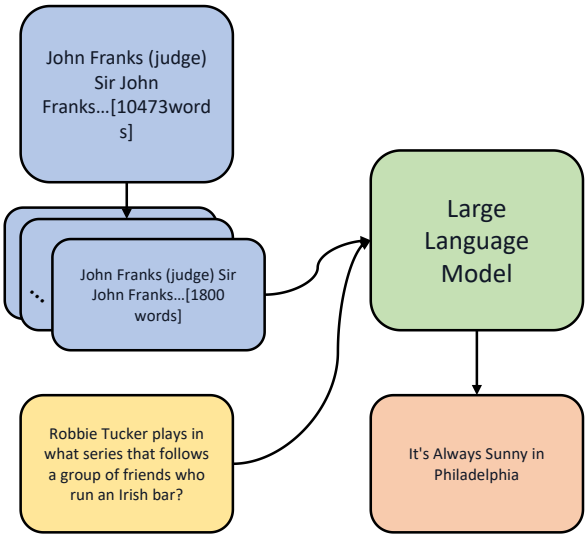

Figure 1: A case study of **AESS** on open domain QA task. We first segment excessively lengthy documents into multiple shorter contexts, to ensure that the LLM can accommodate some of them, and generate responses based on questions and shorter contexts.

LLMs typically use Transformer (Vaswani et al., 2017), but they struggle with long sequences due to quadratic attention complexity. LLMs are, therefore, mostly trained with relatively small context windows (e.g., 4K). Recently, many algorithms have been optimizing this deficiency from a hardware or attention perspective (Dao et al., 2022; Poli et al., 2023; Peng et al., 2023a). Meanwhile, some efforts (Li et al., 2023; Zheng et al., 2023) involve obtaining new models by interpolation (Chen et al., 2023b) and fine-tuning on long texts based on existing base model (Touvron et al., 2023). However, a recent study (Bai et al., 2023) spanning different long text tasks reveals that the above-advanced methods only bring partial improvements or even perform worse than the base model.

Compared to extending the model's original win-

dow, accurately extracting the most relevant information for the task within a limited window length is also a wise alternative solution. In this work, we investigate `attention entropy`, a theoretical metric for measuring the information content of the input, and find that 1) task-relevant information has lower attention entropy, and 2) their position also influences attention entropy. Based on our findings, we introduce a plug-and-play training-free method Attention Entropy Sort and Selection (**AESS**). As shown in Figure 1, the basic idea is to break down the lengthy texts into multiple parallel contexts, and select the most relevant part (estimated by attention entropy) according to the task query.

We experiment our simple method upon four LLMs (including Llama2, LongChat, ChatGLM2, Vicuna) with different context lengths (4k, 16k, 32k), on widely-used LongBench benchmark (Bai et al., 2023). Results show that our AESS could achieve significant and consistent improvements against other retrieval-based methods, for example, Llama2 with our AESS outperforms the strong baseline by an average 9-10% improvement. Further analyses show that our method nicely complements methods that extend LLMs' original context windows, to achieve further lengthy long text comprehension. Our main **contribution** can be summarized as follows: (1) We delve into the concept of attention entropy as a theoretical metric for assessing the information content within an input. (2)AESS is introduced as a plug-and-play method that requires no training. It operates by breaking down lengthy texts into multiple parallel contexts and selecting the most relevant portion, determined by attention entropy, based on the task query. (3)The results indicate that AESS consistently and significantly improves performance compared to other retrieval-based methods.

2 Methodology

2.1 Problem Definition

We formalize the templates of long context tasks as follows: Given the instruction, context, and task-specific input tuple (I, C, T), the model is expected to give the output O. For instance, in a QA task, the instruction I would ask the model to answer the question T according to the context C, which refers to the long document. Generally speaking, I, T and O tend to be short, while C could be a long sequence of several thousand tokens.

2.2 Attention Entropy

Given a sequence $\{x_1, x_2, \ldots, x_T\}$, the attention paid by the last token on a preceding token j is defined by

$$a_{Tj} = \frac{\exp\left(Q_T^\intercal K_j / \sqrt{d}\right)}{\sum_{i=1}^{T} \exp(Q_T^\intercal K_i / \sqrt{d})}$$

Here a_{Tj} is the normalized attention distribution ($\sum_{j=1}^{T} a_{Tj} = 1$). We define the entropy of attention in every single layer k by the entropy value of the last row in the attention matrix.

$$\text{Entropy}_k = \sum_{j=1}^{T} a_{Tj}^k \log a_{Tj}^k$$

We designed three sets of experiments following the setup of multi-document question answering in previous work (Liu et al., 2023). The model inputs are (i) a question to answer and (ii) k documents (e.g., passages from Wikipedia), where exactly one the documents contains the answer to the question and $k-1$ "distractor" documents do not. Performing this task requires the model to access the document that contains the answer within its input context and use it to answer the question:

- The first set involves comparing a document with the correct answer to other distracting documents. The template (I, C, T) remains unchanged except for the context C, which can be a golden document or other distracting documents. We observed that the attention entropy for the golden document is lower than that for distracted documents.

- The second set includes combining a golden document and four other distracted documents as context C, positioning the golden document at the beginning, end, and middle.

- In the third set, we chose the context C from the second set where the golden document is placed in the middle of distracting documents as a comparison. We replaced distracting documents with ones composed of random words or repeated instances of 'an' of the same length to observe changes in attention distribution under different conditions.

The detailed setup and the results are shown in Figure 2. We can draw some conclusions:

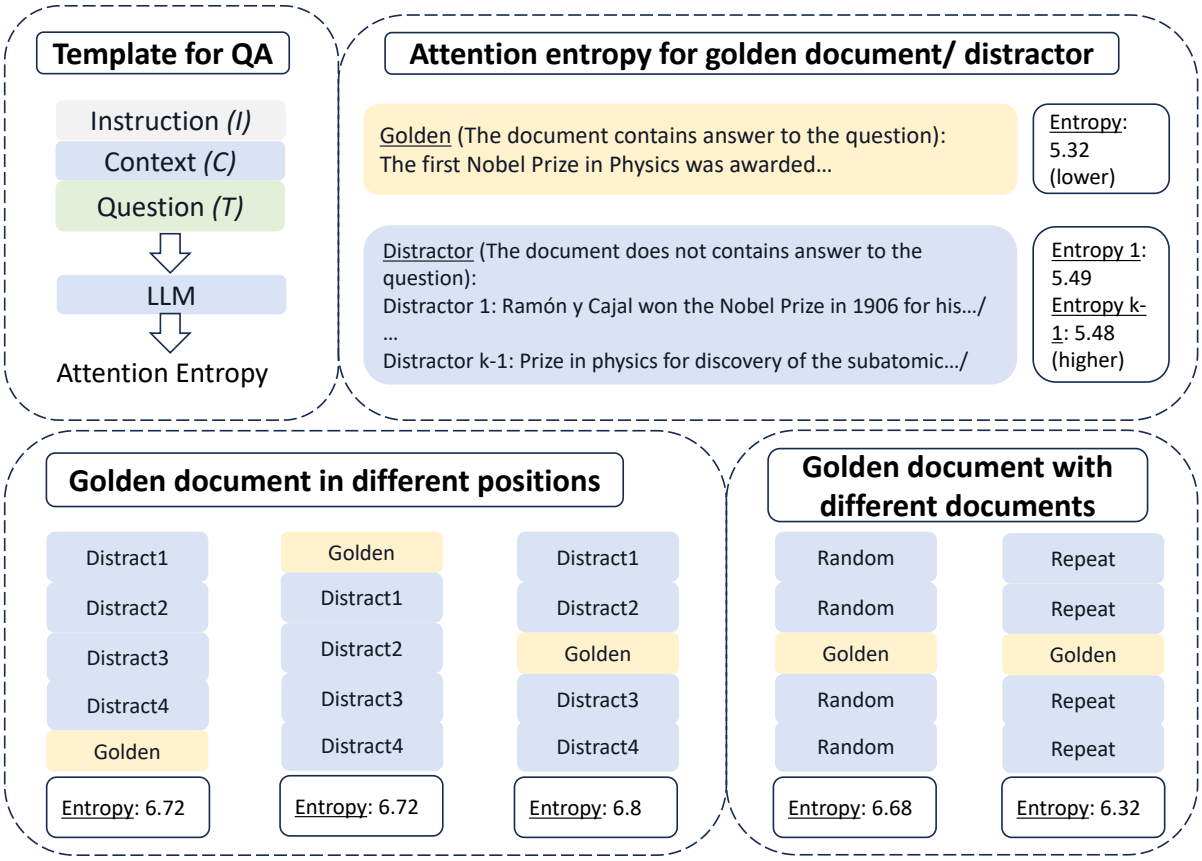

Figure 2: The details of three sets of attention entropy experiments

Attention entropy increases with document length According to Figure 2, we observe that with an increase in the number of documents, attention entropy also increases. By the experiment results, we could find that the attention distribution a_{Tj} is flattened. When the document's length becomes n times longer, we assume the original distribution a_{Tj} is uniformly divided into n parts. The new distribution a'_{Tj} follows $\sum_{i=1}^{nT} a'_{Tj} = \sum_{i=1}^{nT} \frac{a_{Tj}}{n} = 1$, we could calculate the attention entropy:

$$-\sum_{i=1}^{nT} a'_{Tj} \log a'_{Tj} - (-\sum_{i=1}^{T} a_{Tj} \log a_{Tj}) = \log n$$

In real-world scenarios, as the document becomes n times longer, the attention distribution will not be evenly spread out but dispersed with an approximate result. As shown in Figure 2, with a document length five times larger, the attention entropy essentially satisfies $\log n$. However, the specific situation will vary depending on the content of the document.

Position of relevant information influences attention entropy As shown in figure 2, when the golden document's position changes, we observe that when the golden document is at the very beginning or the very end of context C, the attention entropy values are close. However, when the golden document is in the middle of context C, the attention entropy values are higher compared to the first two groups. The golden document contains information most relevant to the task-specific input T, which in the case of multi-document QA includes answers to the questions. These results suggest that when relevant information appears in the middle of the context, it's more challenging for the model to concentrate its attention on the relevant information. The model's attention is more concentrated at both ends. Placing some distracting documents that are less relevant to the question, which was demonstrated to have higher attention entropy in the previous set of experiments, could consequently lead to this outcome.

LLM could pay attention to relevant information LLM tends to ignore meaningless text and pay more attention to the relevant information. In

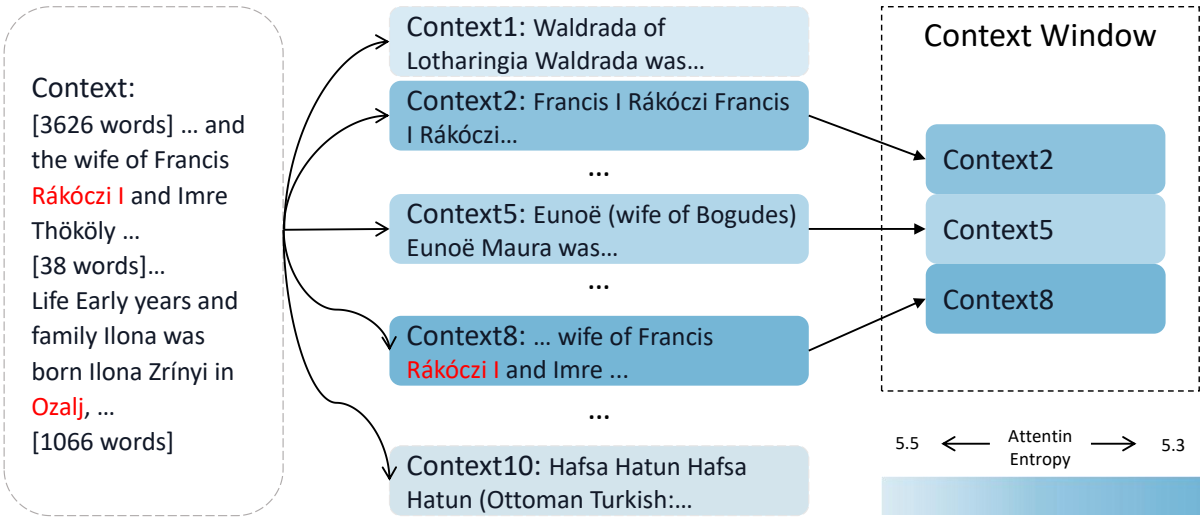

Figure 3: The detailed progress of retrieving the low attention entropy information,

the first set of experiments, when distracting documents surround the golden document, the model's attention might be more scattered as distracting documents contain semantic information, and the model tends to focus on those documents as well. In the second set of experiments, where distracting documents are replaced by random words, the model might concentrate more on the golden document, although some of the random words could still have semantic relevance. In the third set of experiments, where random words are repeated 'an', the model places the primary attention on the golden document, resulting in lower attention entropy.

Attention in different heads have similar entropies In our experiments, we find that attention in different heads has similar entropy while we input the same template. However, for different layers, we find that the entropy in each layer varies.

2.3 Methodology

Implementation Based on the analysis above, we propose our method, attention entropy sort and selection. This method does not require fine-tuning. Our method utilizes the large model itself to retrieve the most relevant information fragments when the context window is restricted., achieving the length generalization of LLM. Moreover, this method is task-agnostic, and different tasks only need different prompts to achieve their retrieval.

As shown in Figure 3, we divide the document into n segments to create shorter contexts. Each segment is then placed into the context of our template (I, C, T) for the calculation of LLM's attention entropy. We sort the attention entropy of the short contexts, and based on the LLM's context window size, select the ones with the lowest entropy. If the original document has an order, such as in multi-turn dialogues or single documents, we maintain the original order when connecting these contexts. For independent short contexts, we connect them in the order of their entropy, from lowest to highest.

As the result mentioned, we use the attention in the first head and calculate the average entropy of all the layers as our final attention entropy.

Context Splitting Since AESS uses the entropy of the LLM as a similarity measure between the prompt and context and focuses on some specific contexts, we need to divide the document D into multiple short contexts. The way to divide the document D depends on the task. We divided the document into some short contexts for the long document QA. For the multi-turn dialogue, we simply adopt a round of users talking, like 'USER:...' + 'ASSISTANT:...' + 'USER:...'. After utilizing our AESS , we merge the overlapping part for semantic coherence. An analysis of the effect of context splitting is shown in the session 5.5.

3 Context Retrieval

Previous work (Liu et al., 2023) shows that model performance degrades as the contexts grow longer, indicating that models struggle to retrieve and use relevant information from long input contexts. Specifically, the model performs best when relevant information appears at the beginning or end of

the input context. Our method, sorting by attention entropy, can bring the information that the model considers most relevant to the forefront, resulting in an accuracy improvement of multi-document QA (The setup follows session2.2) ranging from 9% to 12%.

AESS enhances accuracy as a good retriever The table compares three scenarios with different numbers of retrieved documents (10, 20, 30) to understand the impact on QA accuracy. The baseline accuracy percentages for the Llama method are 59.6%, 57%, and 52.7% for 10, 20, and 30 retrieved documents, respectively. This indicates a general decrease in accuracy as the number of retrieved documents increases. Our method shows improved accuracy over the baseline, with percentages of 66.8%, 62.2%, and 55.1% for 10, 20, and 30 retrieved documents. Sorting appears to enhance accuracy, and the trend suggests a relative improvement across the different document retrieval amounts. Truncation at half of the sorting context shows a slight decrease in accuracy compared to the whole context but remains higher than the baseline. This indicates that as long as the retrieved content is relevant to the question, even a small amount of text can answer the question correctly.

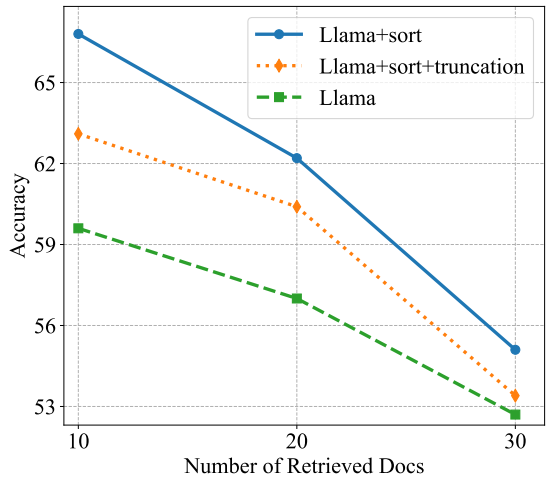

Figure 4: Accuracy for QA experiments with different methods and varying numbers of retrieved documents. As the total text length increases, there is a decrease in the accuracy. Documents sorted using Attention Entropy can generally improve accuracy in this trend. Even when truncating the remaining half of the length, the sorted documents show improvement compared to the baseline.

Larger scale model could retrieve the relevant information better In this study, the performance of three different scales of the Llama-2 model—7B, 13B, and 70B—was evaluated. As presented in Table 1, the baseline exhibits a clear trend of improved accuracy as the scale increases. This suggests that the model's ability is positively influenced by the scale of the underlying architecture. Across all scales, our method has a notable increase in accuracy compared to the baseline, with larger model scales yielding greater improvements. Therefore, our method further enhances the model's ability to extract relevant information across different scales.

	Accuracy		
Model	7B	13B	70B
Llama	57	60.4	67
Llama+sort	62.2	66.6	75.4
$+\Delta$	5.2	6.2	8.4

Table 1: The accuracy of multi-document QA in the different scales of Llama-2 model. Δ calculates the accuracy difference w/wo our method

4 Benchmarks

4.1 Setup

To evaluate the ability of the model to understand the long context, we still evaluate the benchmark in a zero-shot and the template (I, C, T) depends on the tasks. For some baseline models, the input length may surpass the maximum context length, we randomly truncate a window of the context length. As to our method, the implementation follows session 2.2 During generation, we use Top-K sampling.

4.2 Model

5 Results

5.1 Benchmark Results

We evaluate 4 LLMs which are optimized for chat, including Llama2-7B-chat-4k (Touvron et al., 2023), LongChat-v1.5-7B-32k (Li et al., 2023), ChatGLM2-6B-32k (Du et al., 2022; Zeng et al., 2023), and Vicuna-v1.5-7B 16k (Zheng et al., 2023). ChatGLM2-6B-32k is trained based on ChatGLM2-6B, with a 32k context length during alignment and position interpolation (Chen et al.,

Model	1-1	1-2	1-3	2-1	2-2	2-3	3-1	3-2	AVG
Llama2	7.6	15.2	17.2	11.1	16.9	5.3	52.5	4.6	16.3
Llama2+BM25	18.1	19.7	29.8	29.5	22.8	9.0	77.8	29.9	32.1
Llama2+MEMWALKER	**22.2**	**22.5**	**38.4**	28.5	31.4	10.4	78.7	29.0	32.6
Llama2+Ours	21.3	19.8	36.9	**33.5**	**33.0**	**14.2**	**80.7**	**35.6**	35.4

Table 2: Results (%) on single-doc QA, multi-doc QA and summarization tasks. 'AVG' is computed by the macro-average over major task categories.

2023b). LongChat-v1.5-7B-32k and Vicuna-v1.5-7B-16k are fine-tuned from Llama2-7B, with supervised fine-tuning and linear RoPE scaling.

5.2 Datasets

Dataset	ID	Avg len	Metric
Single-Document QA			
NarrativeQA	1-1	18,409	F1
Qasper	1-2	3,619	F1
MultiFieldQA-en	1-3	4,559	F1
Multi-Document QA			
HotpotQA	2-1	9,151	F1
2WikiMultihopQA	2-2	4,887	F1
MuSiQue	2-3	11,214	F1
Few-shot Learning			
TriviaQA	4-1	8,209	F1
SAMSum	4-2	6,258	Rouge-L

Table 3: An overview of the dataset statistics in LongBench. 'Avg len' (average length) is computed using the number of words.

We assess the AESS 's performance on the LongBench benchmark (Bai et al., 2023), comprising of 8 English tasks: NarrativeQA (Kočiskỳ et al., 2018), Qasper (Dasigi et al., 2021), MultiFieldQA, HotpotQA (Yang et al., 2018),2WikiMultihopQA (Ho et al., 2020),MuSiQue (Trivedi et al., 2022),TriviaQA (Joshi et al., 2017), SAMSum (Gliwa et al., 2019). The details of average length and evaluation metric is shown in the table 3

Table 2 report the performance (%) on datasets listed in Table 3. Models benefit from scaled positional embedding and continued training on longer context, as ChatGLM2-6B-32k obtains relative improvement of 44%. But LongChat-v1.5-7B-32k does not exhibit a significant overall improvement on these tasks.

AESS As shown in Table 2, we found that AESS provides a tremendous improvement in various tasks on LongBench when using Llama2 with a 4k context window as the baseline for handling long texts. Compared to other methods without any extra model, our method allows retrieval of the most relevant window based on the question and current output.

Other methods To fairly compare our method without extra parameters, we choose BM25 (Robertson and Zaragoza, 2009) and MEMWALKER (Chen et al., 2023a). We find that the MEMWALKER performs well on the single document task due to their ability to summarize the text. Our method may suffer semantic incoherence during the context splitting. When it comes to the multi-document task, our method overperforms with the other methods.

5.3 AESS is model-agnostic

Table 4 provides a comparative analysis of different models (LongChat, ChatGLM2, Vicuna) in multi-document QA tasks. We can observe that across different models, our method has achieved improvements compared to the baseline. The standout performer in the table is the LongChat, consistently surpassing the baseline. This improvement is evident across all three scenarios, with accuracy rates increasing from 31.5% to 32.5%, 20.6% to 21.3%, and 9.7% to 10.5%. Similarly, the ChatGLM2+Sort variant outperforms the base ChatGLM2 model, showcasing the positive impact of incorporating a sorting strategy. On datasets 2-3, we noticed that the performance of several baseline models was mediocre, but with the inclusion of our method, there were significant improvements.

5.4 Passkey Retrieval

The passkey retrieval is a task from (Mohtashami and Jaggi, 2023) that measures a model's ability to retrieve a simple passkey (i.e., a five-digit number) from amongst a large amount of otherwise meaningless text. With our method, both 7b and

Model	2-1	2-2	2-3
LongChat	31.5	20.6	9.7
LongChat+Sort	32.5	21.3	10.5
ChatGLM2	22.4	20.1	6.1
ChatGLM2+Sort	24.6	20.1	8.1
Vicuna	25.3	20.8	9.8
Vicuna+Sort	25.4	21.2	10.1

Table 4: Different models' result (%) on multi-doc QA.

13b models fine-tuned using YaRN at 64k context size achieve the ability to process 128k input texts remaining essentially unchanged accuracy. Meanwhile, we have achieved steady accuracies by inputting documents of length 48k in context windows of LongLoRA with context window sizes of 8k, 16k, and 32k respectively. We show detailed results in table5.

Model	Context Window	Passkey Context	Accuracy
YaRN-7B	64k	64k	96.3%
YaRN-7B+Ours	64k	128k	96.2%
YaRN-13B	64k	64k	97.5%
YaRN-13B+Ours	64k	128k	97.6%
LongLoRA-8k-ft+Ours	8k	48k	98.9%
LongLoRA-16k-ft+Ours	16k	48k	99.1%
LongLoRA-32k-ft+Ours	32k	48k	99.1%

Table 5: Passkey retrieval performance of YaRN and LongLoRA

5.5 Analysis of Context window

The provided table presents results for different window sizes in the context of three tasks: single-document question answering (QA), multi-document QA, and summarization. For a single sentence, the values are lower than other wider context windows. The result indicates that using a single sentence as the window length for truncation disrupts semantic coherence, and the model struggles to accurately answer questions based on incoherence context. For 10 sentences, the results with higher percentages across all comparisons suggest that a context window to cover 20 sentences enables our method to achieve local optimality. A fixed 200 or 500-token count as context also provides good results, although slightly lower than the 10-sentence or 20-sentence context respectively. This is because truncating text based on the number of tokens may result in incomplete sentences, and connecting them with the subsequent context can alter the intended meaning of the text.

Context window	1-1	2-1	3-1
1 sentence	5.3	10.4	60.3
10 sentences	20.6	31.4	77.9
200 tokens	19.3	27.5	74.2
20 sentences	21.1	32.5	80.1
500 tokens	19.8	28.6	74.3

Table 6: Different window size results (%) on the first dataset of single-doc QA, multi-doc QA, and summarization task.

6 Multi-turn dialogue

One of the applications of our method is multi-turn dialogue. During the multi-turn dialogue, the text is longer with the time increase. However, the context window always has a length restriction. We might retrieve the most relevant history chats to respond.

6.1 Implementation

We form that the dialogue follows 'USER: Q_1', 'ASSISTANT: A_1', 'USER: Q_2', ... There are many turns t and the whole dialogue history can be considered a long document. For new turn $t+1$ generation, We set Q_{t+1} as the T in the template (I, C, T) to retrieve, and the context C is the previous t-turn chat history. Like the figure3, we select the most relevant chats by the attention entropy, and the number of chats depends on the context window.

6.2 Case study

As shown in the figure 5, the user asks the model to generate a passage about the sunset. After several rounds of dialogue, approximately at intervals of 4k tokens, the user asks the model to generate a passage about the night. After further conversation, the accumulated text reaches around 8k tokens, exceeding the context windows of Llama-2. If a regular sliding window is used, the passage about the sunset would have been forgotten. We use the user's current text as a template T to retrieve the previous dialogue history. In this case study, the model's response can combine information from the two generated passages.

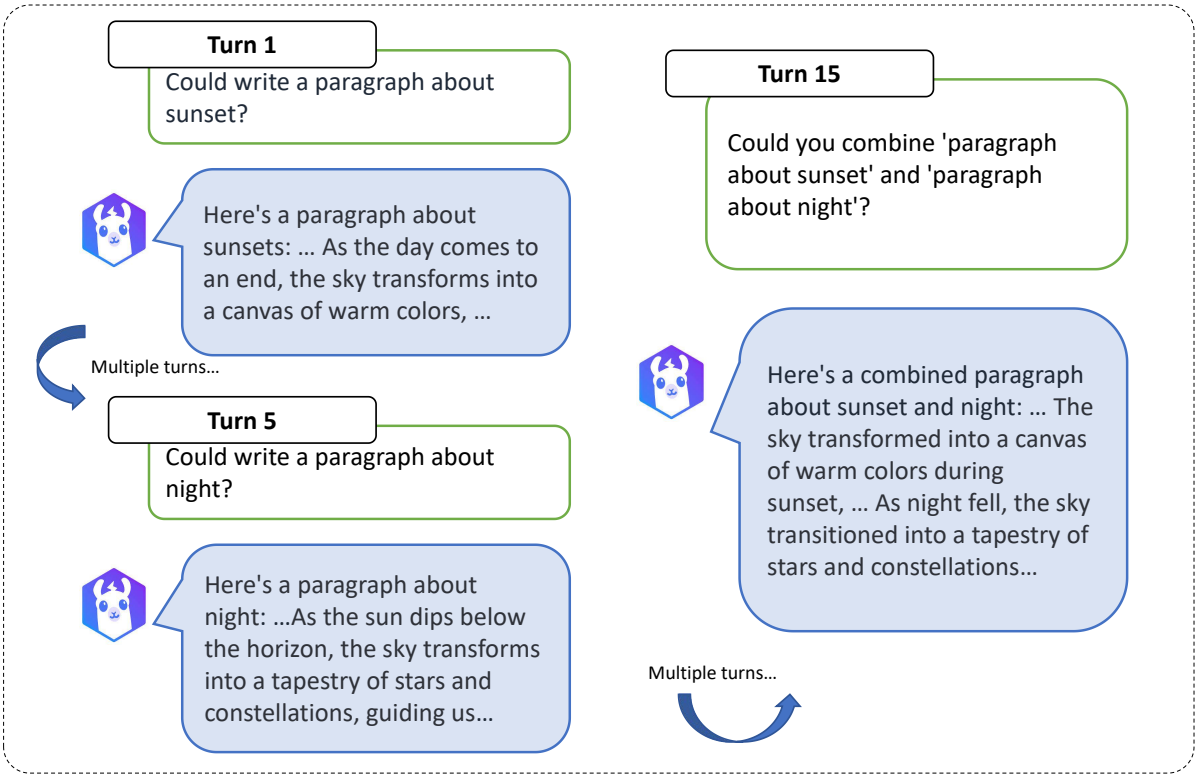

Figure 5: A case study of multi-round dialogue. AESS could retrieve relevant chats

7 Related Work

Our study is highly relevant to two types of researches:

7.1 Long-context Language Models

Many popular lines of methods that aim to tackle challenges in long text modeling, including the high runtime and the catastrophic forgetting phenomenon. A series of studies focus on Transformer variants with modifications like recurrence and memory (Dai et al., 2019; Rae et al., 2020; Wu et al., 2022; Martins et al., 2022; Bulatov et al., 2022; Orvieto et al., 2023; Liang et al., 2023; Zhou et al., 2023), factorizing attention into computationally less intensive approximations (Beltagy et al., 2020; Zaheer et al., 2020), or low-rank approximations (Wang et al., 2020; Peng et al., 2021). Dao et al. (2022) instead provide a faster exact attention by reducing CUDA kernel calculations. Separately, directly replacing attention with convolution and/or linear RNNs, e.g., in RWKV (Peng et al., 2023a), S4 (Gu et al., 2022), or Hyena (Poli et al., 2023). TRAMS(Yu et al., 2023) and H2O(Zhang et al., 2023) are selecting the most important tokens and putting them in the contexts. The representation during this process may be cracked.

7.2 Encoder Retrieval

Izacard and Grave (2021) propose Fusion-In-Decoder for encoder-decoder fine-tuning. The method was applied to open-domain question answering in order to leverage retrieved passages. Specifically, each retrieved supporting passage is encoded by bidirectional encoders. Then the decoder performs conventional attention over the concatenation of the representations of passages. Xu et al. (2024) utilize encoder-based retriever to extract the texts. In comparison, we focus on in-context learning with decoder-only models (such as GPT), without fine-tuning the original model parameters.

8 Conclusion

In this work, we introduced AESS : A simple approach for allowing any off-the-shelf LLM to broaden the scope of text it can access during inference. We showed the effectiveness and universality of our AESS in the single-Doc QA and Multi-Doc QA tasks which potentially enabled LLMs to handle long documents and extended conversations without the risk of context truncation. Further analyses show that our method can nicely complement the long-context LLMs.

Limitations

Despite the promising results, the current method still has some limitations. In the single-document QA task, the semantic incoherence may occur since the text splitting.

Ethics Statement

We place great importance on ethical considerations and adhere strictly to the ACL Ethics Policy.

References

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

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

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. 2022. Recurrent memory transformer. In *NeurIPS*, volume 35, pages 11079–11091.

Howard Chen, Ramakanth Pasunuru, Jason Weston, and Asli Celikyilmaz. 2023a. Walking down the memory maze: Beyond context limit through interactive reading. *arXiv preprint*.

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

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *arXiv preprint*.

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 *NAACL*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *ACL*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *EMNLP-IJCNLP*.

Tanya Goyal and Greg Durrett. 2020. Evaluating factuality in generation with dependency-level entailment. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3592–3603, Online. Association for Computational Linguistics.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. In *ICLR*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *ICLR*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *EACL*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Ehsan Kamalloo, Nouha Dziri, Charles Clarke, and Davood Rafiei. 2023. Evaluating open-domain question answering in the era of large language models. In *ACL*.

Tomáš Kočiskỳ, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *TACL*.

Dacheng Li, Rulin Shao, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. 2023. How long can open-source llms truly promise on context length?

Xinnian Liang, Bing Wang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. 2023. Unleashing infinite-length input capacity for large-scale language models with self-controlled memory system. *arXiv preprint*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *arXiv preprint*.

Qingyu Lu, Baopu Qiu, Liang Ding, Kanjian Zhang, Tom Kocmi, and Dacheng Tao. 2023. Error analysis prompting enables human-like translation evaluation in large language models: A case study on chatgpt. *arXiv preprint*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *ACL*.

Pedro Henrique Martins, Zita Marinho, and André FT Martins. 2022. ∞-former: Infinite memory transformer. In *ACL*.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *arXiv preprint*.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. 2023. Resurrecting recurrent neural networks for long sequences. *arXiv preprint*.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Bolun Wang, Johan S. Wind, Stanislaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. 2023a. Rwkv: Reinventing rnns for the transformer era. *arXiv preprint*.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. 2021. Random feature attention. In *ICLR*.

Keqin Peng, Liang Ding, Qihuang Zhong, Li Shen, Xuebo Liu, Min Zhang, Yuanxin Ouyang, and Dacheng Tao. 2023b. Towards making the most of chatgpt for machine translation. In *Findings of EMNLP*.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *AKBC*.

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. In *ICML*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, Chloe Hillier, and Timothy P Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *ICLR*.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *arXiv preprint*.

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen tau Yih. 2023. REPLUG: Retrieval-augmented black-box language models. *arXiv preprint*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

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*.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *TACL*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NeurIPS*.

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

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing transformers. In *ICLR*.

Peng Xu, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Retrieval meets long context large language models. *arXiv preprint*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Haofei Yu, Cunxiang wang, Yue Zhang, and Wei Bi. 2023. Trams: Training-free memory selection for long-range language modeling. *arXiv preprint*.

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. *NeurIPS*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. 2023. Glm-130b: An open bilingual pre-trained model. In *ICLR*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint*.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. 2023. H_2o: Heavy-hitter oracle for efficient generative inference of large language models. *arXiv preprint*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint*.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Peng Cui, Tiannan Wang, Zhenxin Xiao, Yifan Hou, Ryan Cotterell, and Mrinmaya Sachan. 2023. Recurrentgpt: Interactive generation of (arbitrarily) long text. *arXiv preprint*.