You Only Use Reactive Attention Slice When Retrieving From Long Context

Yun Joon Soh
University of California, San Diego
yjsoh@ucsd.edu

Hanxian Huang
University of California, San Diego
hah008@ucsd.edu

Yuandong Tian
FAIR At Meta
yuandong@meta.com

Jishen Zhao
University of California, San Diego
jzhao@ucsd.edu

Abstract

Retrieval-Augmented Generation is a powerful method for enhancing language models (LMs), but existing retrieval techniques are limited. Embedding-based methods are often inaccurate due to their reliance on lexical similarity, while neural retrievers are computationally expensive to train.

To overcome these issues, we introduce You Only Use Reactive Attention slice (YOURA), a training-free and fine-tuning-free attention-based retrieval technique. When retrieving, YOURA uses a novel **reaction score** heuristic, which quantifies how an LM's self-attention "reacts" to a user query. We also propose a sentence extraction algorithm to efficiently preprocess the context.

Evaluations on three open-source LMs using the LongBench and BABILong datasets show YOURA's effectiveness. Our framework improves QA task accuracy by up to 15% and inference throughput by up to 31% compared to embedding-based retrieval.

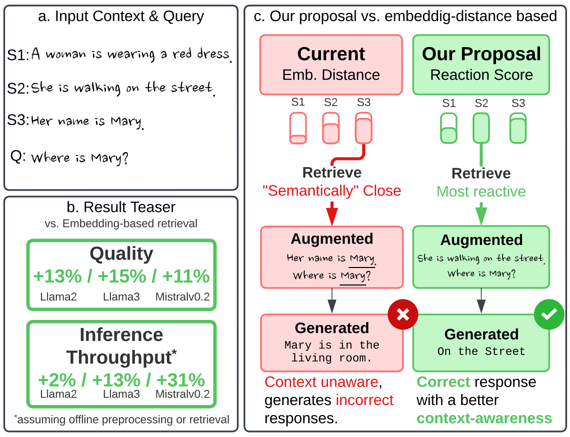

Figure 1: YOURA improves the retrieval quality and inference throughput by retrieving only the "reactive" sentences.

1 Introduction

Language Models (LMs) are integral to natural language processing tasks, whereas a limited context window size remains a critical bottleneck for complex tasks. To address the issue, recent studies explored fine-tuning of pre-trained model (Chen et al., 2023b; Mangrulkar et al., 2022), advanced attention (Xiao et al., 2024; Zhang et al., 2023), and Retrieval-Augmented Generation (RAG) techniques. However, fine-tuning requires extra computational power for each pre-trained model and is often tailored to individual tasks or datasets. Advanced attention mechanisms conceptually extend the context window by selectively choosing a subsequence of tokens for attention, but risk dropping key information.

RAG promises both reduced computational requirements during inference (Luo et al., 2024) and improved accuracy by eliminating irrelevant information (Catav, 2023), emphasizing the importance of retrieval quality. The current RAG techniques typically address the long context challenge by splitting text into smaller chunks and retrieving a subset of chunks that are semantically similar to the query. The semantic similarities are computed via probabilistic retrieval framework, distance measurement within pre-trained embedding space, or neural retriever.

Despite the wide adoption of embedding-based retrieval (e.g., vector databases), it often suffers from a low accuracy for two reasons as illustrated in Figure 1: (1) the common words between a query and a text chunk artificially reduce the semantic distance, making them appear more similar and (2) alphabetically different but semantically identical words can unnecessarily increase the semantic distance. For instance, given a query "Where is Mary?", an embedding-based retriever might incorrectly prioritize a sentence "Her name is Mary" (due to the shared word Mary) to the key sentence "She is at home", leading to an inaccurate retrieval. On the other hand, recent works on neural retriever including dense retriever, reranker, multi-turn re-

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 20665–20686
November 4-9, 2025 ©2025 Association for Computational Linguistics

Long Context QA Task	Key Traits			Benefits		Trade-Off
Approach	Retrieval Heuristics	Context-Aware Retrieval	Retrieved Size	Inference Accuracy	Inference Performance	Preprocessing Overhead
Probabilistic Retrieval	Bag-of-word Frequency	No	Medium	Okay	Okay	**Low**
Embedding Vector Space	Cos. Similarity	No	Medium	Okay	Okay	Medium
Truncate-Middle	Discard Middle	No	Medium	Good	Okay	**Low**
LongLLMLingua	Perplexity	Yes	Small	Good	Fast	High
Proposal: YOURA	**Reaction Score**	**Yes**	**Small**	**Better**	**Fast**	High

Table 1: Comparison of Retrieval Approaches on Long Context QA.

triever, exhibit high computational cost to train a model with large scale training data (Sachan et al., 2021), and often lacks interpretability (Krishna et al., 2022) or robustness (BehnamGhader et al., 2022; Dai et al., 2024).

To address these challenges, we propose You Only Use Reactive Attention slice (YOURA), a training-free, fine-tuning-free, attention-based retrieval strategy. At its core, YOURA offers a **novel theoretical framework for attention-based retrieval**, which interprets attention values as the probability of a token being importance. From this insight, the importance of context chunks can be derived through statistical approaches: (i) employing a joint probability to indicate the importance of a chunk within an overall context, and (ii) using a likelihood ratio to determine the chunk-query relevance. Because of its generic interpretation of attention scores, YOURA can be applied to various off-the-shelf pre-trained LMs to achieve improved QA task accuracy at a higher inference throughput by using only the relevant information.

YOURA consists of two main steps: preprocessing and retrieval. During preprocessing, YOURA first computes the initial self-attentions from the context alone, intentionally leaving out the query. Unlike traditional chunking mechanisms (Kamradt, 2024), YOURA's approach is to split the context token sequence into sentences to allow for more granular retrieval. This is a non-trivial task due to subword tokenization, which is handled by a dedicated sentence extraction algorithm (Section 3.4). For retrieval, YOURA measures the self-attention shifts that occur when the query is appended to the context. This shift is captured in a **reaction vector**, which quantifies how the context "reacts" to the presence of the query. Then a **reaction score** is calculated as the average of a slice of this vector, serving as a measure of semantic similarity to the query. The algorithm greedily retrieves sentences based on their individual reaction score.

YOURA achieves (1) higher retrieval quality, (2) improved generation quality, and (3) increased inference throughput. These benefits are a result of several factors. (a) The reaction score is measured holistically (rather than independently for each sentence), enabling context-aware retrieval. (b) This context-aware retrieval filters out distracting information, improving generation quality. (c) LLM serving platforms process fewer tokens, which increases inference throughput.

While YOURA's main trade-off is its preprocessing overhead, it can be amortized over multiple requests that use the same preprocessed context. For single requests, YOURA still reduces overhead by reusing the KV cache during its initial and reacted attention vector computation.

Evaluation on LongBench (Bai et al., 2024) shows that YOURA improves the QA accuracy by 15% and vLLM (Kwon et al., 2023) serving throughput by 31% for the QA task when compared against the embedding-based retrieval. For the Needle-In-A-Haystack task evaluated with BABILong (Kuratov et al., 2024) dataset shows a 25% improved accuracy across five subtasks using off-the-shelf Llama3.

This paper contributes the following:

- We propose **You Only Use Reactive Attention slice (YOURA)**, a novel training-free and fine-tuning-free attention-based retrieval framework. To our knowledge, YOURA is the first to achieve efficient and accurate long-context retrieval by directly leveraging attention slices.
- We propose a **sentence extraction algorithm** that efficiently slices a token sequence into per-sentence subsequences with 94% accuracy.
- We evaluate YOURA with three open-source pre-trained models (Llama2-7B, Llama3-8B, and Mistralv0.2-7B) on nineteen different subsets from the LongBench (Bai et al., 2024) and BABILong (Kuratov et al., 2024) dataset for comprehensive understanding of its benefits.
- We analyze the YOURA's runtime overhead and show the overhead amortization potentials.

2 Related Work

Limited context window size of language models (LMs) is a well-acknowledged problem and various approaches such as longer context window sized models, novel attention mechanisms, fine-tuned models, context compression techniques, in-context learning, test-time learning, and Retrieval-Augmented Generation (RAG) have been proposed. In this section, we discuss retrieval related techniques, and list other techniques in appendix.

Retrieval-Augmented Generation. Retrieval-Augmented Generation (RAG) retrieves relevant context chunks from the preprocessed knowledge base and provides the following benefits: processes inputs longer than the language model's context window size, improves generation quality by excluding distractful information, and reduces computational cost with fewer input tokens to autoregressively generate from. Prior works integrate RAG with language models for question answering with long documents (Stelmakh et al., 2022; Qian et al., 2024) and in open-domain (Giorgi et al., 2022; Izacard and Grave, 2021). Furthermore, language models adopted retrieval at various stages such as pretraining (Wang et al., 2023; Izacard et al., 2023; Borgeaud et al., 2022), fine-tuning (Jiang et al., 2022; Zhang et al., 2024) and inference stage (Khandelwal et al., 2019).

Probabilistic Retrieval Framework. BM25, a probabilistic retrieval framework, was widely adopted as a relevance measurement of two texts (Robertson et al., 1995). Two texts yield high relevance if they have a high frequency of identical word sequences, with some adjustment to mitigate false positive scenarios such as a repetitive phrase or common phrase like "is a".

Neural Retriever. The advent of neural networks inspired various techniques such as dense retriever, or multi-turn retrievers. One popular approach is to take a predefined embedding vector space, a vector representation of the query, and a list of candidate information chunks as input, and outputs a numerical value per chunk representing the chunk's query relevance. A pairwise cosine similarity of vectors is a common formula (Reimers, 2019; Douze et al., 2024; Chase, 2022; Liu, 2022) for computing the distance with some alternatives such as dot product (Karpukhin et al., 2020; Khattab and Zaharia, 2020).

LongLLMLingua (Jiang et al., 2023b) observed that the small LM is capable of identifying the key relevant information, and devised a heuristic that ranks the context chunks based on perplexity computed with the small LM. Unlike their approach, which relies on a relatively complex perplexity calculation, our approach has simpler math and reasoning behind the retrieval heuristic design.

3 You Only Use Reactive Attention Slice (YOURA)

Traditional Retrieval-Augmented Generation (RAG) systems typically segment input into discrete chunks, assessing the relevance of each chunk to the query in isolation. This segmented approach can overlook inter-chunk dependencies, leading to potential context fragmentation and a loss of nuanced, cross-chunk relationships. In contrast, YOURA introduces a holistic method that evaluates contextual relevance across the entire input sequence. By computing relevance as a whole, YOURA preserves cross-chunk context, resulting in more accurate and contextually cohesive retrieval. This approach enables YOURA to prioritize relevant chunks not only by direct query similarity but also by their broader contextual alignment within the input, enhancing retrieval precision and relevance.

Design Overview. Figure 2 illustrates an overview of our design. The left side of the figure depicts how a reaction vector is calculated, while the right side shows how this vector is used for retrieval and how the language model then leverages the retrieved sentences to answer a query.

3.1 Problem Statement & Definitions

Annotations. We annotate $\mathrm{Z}^C \in \mathbb{R}^{d\times c}$, $\mathrm{Z}^Q \in \mathbb{R}^{d\times q}$ and $\mathrm{Z}^{CQ} \in \mathbb{R}^{d\times(c+q)}$ as the continuous representation of the context, query-only, and query-appended context, respectively. d is the pre-trained model's hidden dimension, c is the context token count and q is the query token count.

Reaction Vector. We define the **reaction vector** as follows:

$$\mathrm{ReactionVec}(\mathrm{Z^C}, \mathrm{Z^Q}) = \frac{\mathrm{AttnVec}(\mathrm{Z^{CQ}})_{1:c}}{\mathrm{AttnVec}(\mathrm{Z^C})} \quad (1)$$

where division in the equation is element-wise division, and

$$\mathrm{AttnVec}(\mathrm{Z}) = \mathrm{Mean}_{col}(\mathrm{AttnMatrix}(Q, K)) \quad (2)$$

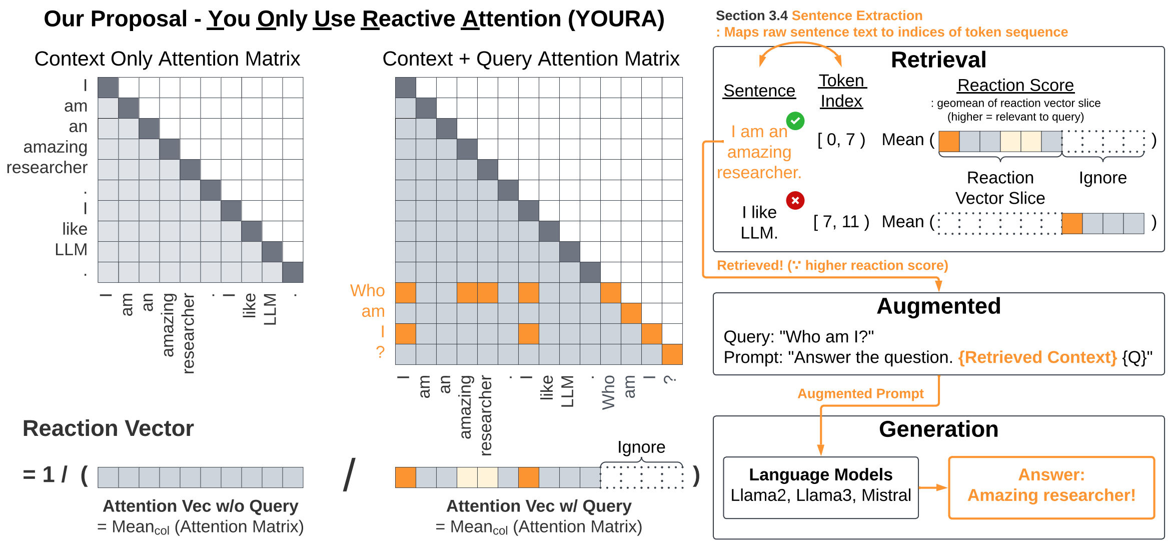

Figure 2: Overview of YOURA and where it is used in the Retrieval Augmented Generation (RAG) with an example (example context: "I am an amazing researcher. I like LLM.", example query: "Who am I?"). The first step is calculating the **reaction vector**, the ratio between the attention vector with and without the query (left side of the figure). The highlighted cells in the attention matrix indicate that the token pair exhibits a relatively high value (e.g., Who vs. I). Once the reaction vector has been calculated, each sentence is assigned a **reaction score**, the mean of a corresponding reaction vector slice. To map each sentence to a token sequence, we propose the sentence extraction algorithm (Section 3.4). The retriever passes on the sentences with high reaction scores to the augmenter. The pre-trained LLM models generate answers using the augmented text, which includes the task-specific prompt, the retrieved context, and the query.

AttnMatrix is the dense attention function before multiplying the value projection and Q, K are the query and key projection of Z, respectively, similar to the description in the "Attention is all you need" paper (Vaswani et al., 2023):

$$\begin{aligned} \text{AttnMatrix}(Q, K) &= \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \\ Q &= \mathrm{Z} \times W^Q \\ K &= \mathrm{Z} \times W^K \end{aligned} \tag{3}$$

Mean$_{col}$ takes a matrix ($[a_{ij}]_{n_1 \times n_2}$) and returns a per-column mean vector:

$$\begin{aligned} \text{Mean}_{col}([a_{ij}]_{n_1\times n_2}) &= \frac{1}{n_1}\left(\sum_{i=1}^{n_1} a_{i1}, ..., \sum_{i=1}^{n_1} a_{in_2}\right) \\ &= \frac{1}{n_1}\left(\sum_{i=1}^{n_1} a_{ij}\right)_{j=1}^{n_2} \end{aligned} \tag{4}$$

For multi-head attention, the reaction vector is averaged across head dimension as well:

$$\text{Mean}_{h,col}\left([a^h_{ij}]_{n_1\times n_2}\right) = \frac{1}{H}\sum_{h=1}^{H}\frac{1}{n_1}\left(\sum_{i=1}^{n_1} a^h_{ij}\right)_{j=1}^{n_2} \tag{5}$$

Note that in practice, the algorithm precomputes the KV cache for the given context and reuses it for efficient attention computation. Thus, the query-appended attention matrix is $[a^h_{ij}]_{q\times(c+q)}$.

Reaction Score Given the reaction vector, we define the **reaction score (rs)** as follows for a vector slice represented with left-right index, $[l, r)$:

$$\begin{aligned} &\text{ReactionScore}(\text{ReactionVector}, l, r) \\ &\quad = \text{geomean}(\text{ReactionVector}[l:r]) \end{aligned} \tag{6}$$

where *geomean* is the geometric mean.

Problem Statement We define retrieval task as finding a non-overlapping chunk set $S = \{(l_i, r_i) \mid 0 \le l_i \le r_i \le c,\ r_i \le l_{i+1}\}$, such that

$$\text{argmax}_S \ \text{AnsQuality}(\text{LLMInf}(\text{concat}(S, q))) \tag{7}$$

where:

- concat(S, q) denotes appending the query q to the retrieved chunks S.
- LLMInf is the language model inference given the query-augmented input.
- AnsQuality judges the generated sequence quality (e.g., F1 Score).

3.2 Interpreting Reaction Vector as Likelihood Ratio

Our analysis relies on a fundamental assumption: **the i-th element of the attention vector is the probability of the i-th token being important.** We refer to this as the ***importance probability***. Based on this assumption, we interpret the reaction vector as a vector of ***importance likelihood ratios*** that arise from the presence of the query tokens. The reaction score, defined as the geometric mean of a likelihood ratio vector slice, can then be interpreted as the likelihood of the normalized joint probability. Ultimately, we use the reaction score to represent the relevance of a vector slice to the appended query.

AttnVec as Probability Vector. Assume $\text{AttnVec}(Z^C)$ and $\text{AttnVec}(Z^{CQ})$ are vectors of importance probabilities:

$$\begin{aligned} \text{AttnVec}(Z^C) &= \{p_1^C, ..., p_c^C\} \\ \text{AttnVec}(Z^{CQ}) &= \{p_1^{CQ}, ..., p_{c+q}^{CQ}\} \end{aligned} \tag{8}$$

where p_i^C and p_i^{CQ} are the probability of i-th token being an important token within the context and context-query concatenation, respectively.

ReactionVec as Likelihood Ratio Vector. Each element within the reaction vector is a ratio: the probability of a token being important in the context-query concatenation versus its probability in the context alone.

$$\begin{aligned} \text{ReactionVec}(Z^C, Z^Q) &= \{r_1, r_2, ..., r_c\} \\ \text{where } r_i &= \frac{p_i^{CQ}}{p_i^C} \end{aligned} \tag{9}$$

r_i indicates whether the i-th token has increased importance in the presence of the query ($r_i > 1$) or is more significant in the context alone ($r_i < 1$).

ReactionScore as Likelihood of Normalized Joint Probability. The reaction score (geometric mean of the reaction vector slice) can be seen as a likelihood of normalized joint probability. The geometric mean is used to mitigate the influence of individual tokens with disproportionately high attention values, which might otherwise lead to the retrieval of noisy or less relevant sentences.

$$\begin{aligned} \text{ReactionScore}(\text{ReactionVec}, a, b) &= (\Pi_a^b r_i)^{\frac{1}{(b-a)}} \\ &= (\Pi_a^b \frac{p_i^{CQ}}{p_i^C})^{\frac{1}{(b-a)}} \\ &= \frac{(\Pi_a^b p_i^{CQ})^{\frac{1}{(b-a)}}}{(\Pi_a^b p_i^C)^{\frac{1}{(b-a)}}} \end{aligned} \tag{10}$$

A value greater than one implies that the token subsequence has greater significance in the query-appended context, highlighting its query-relevance.

3.3 Overhead Reduction via KV Cache Reuse

ReactionVec computation can be optimized by reusing the KV cache generated from the context-only sequence (Z^C) when computing the attention for the query-appended sequence (Z^{CQ}). Since Z^C is a prefix of Z^{CQ}, the Key (K^C) and Value (V^C) matrices for the context can be reused, similar to how K and V are reused during auto-regressive decoding. We discuss the performance gain in Appendix I, Figure 3.

3.4 Sentence Extraction Algorithm

Challenge. Identifying the token index for sentence boundaries in a sequence is challenging due to two main issues: (1) the embedding model used for sentence splitting (e.g., Stanza) often differs from the embedding model of the language model (LLM), and (2) models with large token dictionaries can create sequences where perfect sentence alignment is unachievable. For instance, popular sentence-splitting algorithms like Stanza (Qi et al., 2020) use a distinct vocabulary from models such as Llama3, leading to differences in total token counts across models (Table 12).

A further complication arises with models using extensive token dictionaries, making it challenging to pinpoint exact sentence boundaries. For example, Llama3 may treat sequences like `.T` as a single token. This is beneficial for tasks like code generation but problematic when a period is followed by a sentence starting with "The." Stanza might tokenize this sequence as `.` and `The`, while Llama3 could tokenize it as `.T` and `he`. This discrepancy means an exact sentence boundary for Llama3 may not correspond to a single token index.

Sentence Extraction. To address the challenges, we propose the sentence extraction algorithm. The algorithm takes as input a token sequence from the

LLM tokenizer and a list of sentence strings from an NLP sentence-splitting model (e.g., Stanza (Qi et al., 2020)), and outputs a list of token indices for all sentence boundaries.

The sentence extraction algorithm iteratively estimates sentence boundaries and refines them with a best-effort adjustment strategy. Initially, the algorithm calculates a candidate boundary index based on the cumulative encoded length of previously processed sentences. It then decodes the token subsequence between the **most recent confirmed boundary** (variable m) and the **current candidate index**, comparing this decoded sequence to the target sentence.

Depending on the comparison outcome — either `MATCH`, `INCLUDED`, or `NO-MATCH` — the algorithm adjusts the candidate index incrementally (+1 or -1) until a match is achieved or a termination condition is met. Once a match is confirmed, the algorithm records this candidate in the boundary list. In rare cases where adjustments fail, the algorithm reverts to the original candidate index, ensuring a one-to-one mapping between each target sentence and its boundary token index.

3.5 End-to-end Retrieval Process

The retrieval process begins by calculating the reaction vector for the input context. If the context length exceeds the model's maximum context window size, we split the input into smaller inputs that fit within this window, calculate the attention vector for each chunk, and then concatenate these vectors to form a complete reaction vector. This final reaction vector spans the full length of the input context's token sequence.

To segment the reaction vector by sentence, we apply the sentence extraction algorithm (Section 3.4), which returns a list of sentence boundary indices. Using these indices, the retrieval algorithm slices the reaction vector for each sentence and calculates the geometric mean for each slice, generating a list of reaction scores corresponding to individual sentences. These scores serve as a heuristic for retrieval.

For the actual retrieval step, sentences are added in descending order of reaction score until either the retrieval token budget is depleted or 80% of the total sentence count is reached. As a final step, the retrieved sentences are reordered to match their original positions within the context.

4 Experiments

We evaluate how YOURA improves the answer quality of three pre-trained models on QA tasks, including single-document, multi-document, and the needle-in-a-haystack task. Results for non-QA task are listed in Appendix D.1

Datasets. We evaluate YOURA on the single-document QA, multi-document QA, summarization, few-shot learning, and synthetic task datasets (Table 12). For Needle-In-A-Haystack style evaluation, we used BABILong (Kuratov et al., 2024)'s open-sourced dataset for QA1 through QA5.

Models. We used three open-source LMs: `LLama2-7B-Chat-HF`, `Llama3-8B-Instruct`, and `Mistralv0.2-7B-Instruct`, with context window of 4 K, 8 K, and 32 K tokens, respectively.

For the embedding retrieval baseline, we used the `paraphrase-MiniLM-L6-v2` model. Unlike recent state-of-the-art models such as BGE, E5, and Instructor XL, which are trained on standard QA evaluation datasets, our choice has not been. We selected this model to ensure our baseline results would not be influenced by pre-existing biases learned from these benchmarks, providing a more neutral and reliable comparison.

Quality and Inference Evaluation. For YOURA implementation, we leveraged the HuggingFace (Wolf et al., 2020)'s dense attention and features (e.g., `output_attention` option). LongBench (Bai et al., 2024) evaluation scripts were used for quality assessment. For inference-only throughput measurement, we used vLLM (Kwon et al., 2023) throughput benchmark (v0.5.3) with the default vLLM settings with an exception of setting `ignore_eos` to `false`.

Machine Configurations. We used a single server with two Intel(R) Xeon(R) Gold 5416S, 512G DRAM, and one NVIDIA H100-80G GPU.

Baselines. We compared YOURA against the following baselines to assess improvements in output quality and performance. For each setup, as done with the LongBench, we set the retrieval budget to 3500, 7500, and 31500 for models with 4K, 8K, and 32K context window, respectively.

- **N/A:** No context is passed.
- **BM25:** Bx_y indicates chunking at roughly x tokens and retrieving the top y chunks.
- **Embedding Retrieval:** Ex_y indicates chunking at roughly x tokens and retrieving the top

y chunks. We measure the cosine similarity between each chunk's embedding vector generated by the `paraphrase-MiniLM-L6-v2` model.

- **Truncation:** We use LongBench's prompt along with its context truncation approach: the whole context, if it fits within the pre-trained model's context window size, or the concatenation of the context head and tail with an equal context window budget, otherwise.
- **LongLLMLingua:** LLL indicates the LongLLMLingua (Jiang et al., 2023b) with parameter described in Appendix J.

4.1 LongBench Result

QA Accuracy. Table 2 shows the answer quality using three different open-sourced models (Llama2, Llama3, and Mistralv0.2). We make the following observations: (1) We observe that YOURA showed better overall answer quality than all of the retrieval approaches (BM25 and embedding-based) at a higher retrieval ratio — i.e., retrieving fewer tokens from the long context. (2) In comparison to the truncate-middle approach, YOURA shows better overall quality for Llama2 and Llama3, and is equal for Mistralv0.2. (3) When compared against LLL, YOURA outperforms at single document QA and slightly worse at multi-document QA task.

vs. Truncate-Middle. By dataset types, we observe that YOURA improves multi-document QA more than single-document QA across all tested models. This suggests that Trunc. accidentally filters out the key information for multi-document QA, but the head and tail of a single document QA dataset are often sufficient. In contrast, YOURA properly retrieves key information for both the single-document and multi-document QAs resulting in a better answer quality for both tasks.

For Mistral, which has a large context window size of 32k, truncation rarely happens (as shown in the small retrieval ratio). Without truncation, distractful information is included as part of the augmented context, which results in worse single-document answer quality than YOURA. [1] For multi-document QA, YOURA showed slightly worse quality (-0.56, 2%), because a failure to retrieve all the necessary information across multiple documents results in incorrect answers. Overall, the benefit of single-doc accuracy improvement offsets the multi-document accuracy drop resulting in similar accuracy from a higher retrieval ratio.

[1]The impact of distractful information on output quality is observed in other work (Catav, 2023) as well.

vs. LongLLMLingua. LongLLMLingua outperforms all tested retrieval methods for multi-document QA task. Especially for small retrieval budget (3500 tokens for LlaMA2), LongLLMLingua showed 16% better accuracy than YOURA. However, due to its bad accuracy for single document QA, YOURA has higher overall accuracy than LongLLMLingua.

LM	Retrieval	Ratio Avg	SDoc Avg	MDoc Avg	Avg
Llama2	N/A	INF	13.65	19.30	16.47
	B500_7	3.28	22.82	20.17	21.49
	B50_70	3.28	23.86	23.25	23.56
	E500_7	3.28	17.13	19.57	18.35
	E50_70	3.28	24.15	22.87	23.51
	Trunc.	3.28	25.74	22.34	24.04
	LLL	3.79	21.32	**30.55**	25.94
	YOURA	3.39	**26.88**	26.31	**26.59**
Llama3	N/A	INF	11.59	23.16	17.37
	B500_15	1.57	34.07	33.27	33.73
	B50_150	1.57	32.92	35.83	34.17
	E500_15	1.57	33.33	30.82	32.08
	E50_150	1.57	31.50	35.70	33.60
	Trunc.	1.57	36.83	34.72	35.93
	LLL	1.94	33.05	**38.59**	35.82
	YOURA	1.76	**38.82**	38.44	**38.63**
Mistralv0.2	N/A	INF	7.02	11.47	9.25
	B500_63	1.04	32.29	24.66	29.02
	B50_630	1.04	29.14	23.28	26.63
	E500_63	1.04	26.79	22.06	24.76
	E50_630	1.04	29.14	23.28	26.63
	Trunc.	1.04	32.65	26.62	**29.64**
	LLL	1.08	27.99	**27.79**	27.89
	YOURA	1.33	**33.22**	26.06	**29.64**

Table 2: Accuracy of single and multi-document QA tasks. YOURA shows better quality with a higher retrieval ratio, defined as the ratio of context tokens to retrieved tokens (i.e., context tokens / retrieved tokens). All retrieval methods, except for the no-context baseline, are given the same maximum token budget (4K, 8K, and 32K tokens depending on the model). Note that the results for Mistralv0.2 with LongLLMLingua on certain datasets were excluded due to runtime error (see Table 13 & Appendix J for details).

LM	Retrieval	SDoc Avg	MDoc Avg	Avg
L3	E5M_500_15	32.57	**39.10**	35.83
	E5M_50_150	36.49	37.90	37.20
	YOURA	**38.82**	38.44	**38.63**
M	E5M_500_15	29.42	**26.57**	28.00
	E5M_50_150	30.39	26.14	28.27
	YOURA	**33.22**	26.06	**29.64**

L3: Llama3-8B, M: Mistralv0.2

Table 3: Accuracy of single and multi-document QA tasks compared against E5-Mistral (E5M) embedding model-based retrieval.

E5-Mistral Result. To further validate the effectiveness of our proposed training-free method, we conducted a supplemental experiment comparing YOURA against a strong embedding retrieval baseline, E5-Mistral (Table 3). This model has been trained on a diverse set of datasets, with a known overlap with standard QA benchmarks such as HotpotQA and raw Wikipedia data, which likely contributed to the 2WikiMQA or Musique dataset. While training on these dataset provides E5-Mistral with a clear advantage on multi-document QA datasets, our comparison on Llama3 and Mistralv0.2 showed that YOURA's attention-based retrieval strategy was able to achieve higher performance. This result is significant as it demonstrates that our method, which requires no pre-training or fine-tuning, can outperform a thoroughly trained embedding model, highlighting the robustness and efficiency of our approach.

4.2 Inference Performance

We discuss the impact of retrieval on the actual inference time. To clarify, we measure pure inferencing time where the input is an already retrieved and augmented. LongBench (Bai et al., 2024) takes such an offline approach and we created a similar scenario for the performance comparison in Table 4. To understand the YOURA's runtime overhead when retrieval is performed online, we show the latency breakdown in Appendix I, Figure 3.

QA Inference Performance. Table 4 shows the average vLLM inference throughput (requests per second) **assuming offline truncation/retrieval**, just like the LongBench implementation. We make the following observations. First, YOURA

LM	Retrieval	SDoc Avg	MDoc Avg	Avg Req/Sec
L2	Trunc.	5.25	5.76	5.50
	YOURA	**5.43**	**5.78**	**5.60 (+2%)**
L3	Trunc.	3.23	2.79	3.01
	YOURA	**3.76**	**3.03**	**3.39 (+13%)**
M	Trunc.	1.97	1.36	1.66
	YOURA	**2.54**	**1.82**	**2.18 (+31%)**

L2: Llama2-7B, L3: Llama3-8B, M: Mistralv0.2

Table 4: vLLM performance on single-document and multi-document QA. This table compares the average vLLM throughput (request per second) of Truncation and YOURA on LongBench datasets. A key finding is that YOURA's retrieval of fewer tokens leads to an increase in inference throughput of up to 31%.

improves the overall inference throughput, especially for models with larger context window sizes. This is because the retrieval ratio is larger for models with larger context window sizes. Second, the retrieval ratio is a good throughput estimator. The relative throughput improvements (+2%, +13%, +31%) are on par with the relative retrieval ratio (+3%, +12%, +28%). We conclude that the performance of LLM serving platforms such as vLLM is sensitive to the retrieved context size and underscores the importance of high-quality information retriever.

4.3 BABILong Results

BABILong is a synthetic benchmark that aims to answer a question where the key supporting sentence is hidden in the dataset. We observe the following from Table 5: (1) YOURA improves the performance of off-the-shelf Llama3 accuracy by 25% while it shows 15% worse performance for Mistralv0.2. (2) YOURA demonstrates a bias toward supporting fact retrieval (QA1,2,3), likely benefiting from its holistic context-aware retrieval strategy. However, it shows weaker performance in locating arguments (QA4,5), which may require fine-grained positional sensitivity.

4.4 Sentence Extraction Quality

We evaluate the quality of sentence extraction algorithm by measuring its exact match with the ground-truth sentences. We also compare against delimiter-based baselines.

Ret.	QA1	QA2	QA3	QA4	QA5	Avg
Meta-Llama-3-8B-Instruct, Data Length: 8 K						
N/A*	62	20	11	52	**73**	43.6
YOURA	**75**	**48**	**33**	**53**	63	**54.4**
Mistral-7b-Instruct-v0.2, Data Length: 32 K						
N/A*	45	**7**	12	**54**	**67**	**37**
YOURA	**51**	6	**14**	46	41	31.6

*: leaderboard data

Table 5: BABILong Accuracy. Leaderboard is available on https://hf.co/spaces/RMT-team/babilong and from the July 29, 2024 snapshot.

LM	Accuracy ↑	Edit Distance ↓	NZ Edit Distance ↓
L2	0.943	0.52	2.09
L3	0.925	0.24	1.93
M	0.943	0.54	2.12
Avg	0.937	0.43	2.05

L2: Llama2-7B, L3: Llama3-8B, M: Mistralv0.2

Table 6: Sentence extraction algorithm's accuracy and quantified quality for QA task datasets. Accuracy is measured as the average EM rate (exact match / total # of sentences). Unit for edit distance is characters, thus smaller number means smaller misalignment.

Sentence Extraction Accuracy. Table 6 shows the sentence extraction algorithm's accuracy. We used Stanza (Qi et al., 2020), an open-sourced NLP toolkit, to split raw text into sentences, which we treat as the gold standard (our *target sentence*). Then we ran the algorithm on each tokenizer's output to identify the sentence boundaries within a token sequence. A match is recorded if a decoded token segment (defined by a boundary index) perfectly aligns with a target sentence after cleanup (e.g., stripping whitespace).

Overall the extraction algorithm successfully identifies the sentence boundary for models with varying dictionary sizes: Llama2, Llama3, and Mistralv0.2 tokenizers with 32000, 128256, and 32000 words, respectively. Llama2 and Mistral showed nearly identical match ratios, as both models' tokenizers share a similar vocabulary. Llama3's tokenizer, which uses roughly $\times 4$ more vocabulary, results in fewer total tokens (as shown in Table 12, e.g., the single token `.T`). Due to the rich vocabulary, typos easily blur sentence boundaries, causing many imperfect matches and leading to the smallest overall match quality in the 2-1, 2-2, and 2-3 datasets. In contrast, this rich vocabulary proved beneficial for the 1-1, 1-2, and 1-3, datasets, where many atypical char sequences (e.g., <b>, math formulas) were present and were properly represented by the Llama3 tokenizer.

Comparison Against Baselines. To provide a robust point of comparison, we developed a delimiter-based baseline that slices at common punctuation tokens. This rule-based approach serves as a simple method for token-to-sentence alignment. We evaluated the baselines by calculating the mean of the minimum edit distances between the ground-truth sentences and the sentences generated by the baseline. Because the delimiter-based algorithm often produces a larger number of sentences than the ground truth, we employed an exhaustive approach, finding the minimum edit distance for each ground-truth sentence against all generated sentences. As shown in Table 7, our method significantly outperforms the baselines in both Exact Match and Average of Minimum Edit Distance. The poor baseline performance is attributed to a cascading effect, where a single misalignment near the beginning of a sequence leads to a series of subsequent, unrecoverable misalignments, resulting in near-zero accuracy.

LM	Method	Accuracy ↑	Edit Distance ↓
L3	Delim(`.`)	0.000	80.68
	Delim(`.?!;\n...`)	0.000	44.34
	Ours	**0.925**	**0.24**

L3: Llama3-8B

Table 7: Sentence extraction algorithm outperforms delimiter-based sentence splitting in both the accuracy and quantified quality for QA task. Delimiter-based solutions suffers from cascading misalignments.

5 Conclusion

We presented a novel attention-based sentence retrieval framework that leverages a holistic measurement of sentence-to-query relevance to enhance retrieval accuracy, generation quality, and inference throughput. Our proposed heuristic combines strong theoretical foundations with practical effectiveness for sentence retrieval in long-context QA tasks. While the framework introduces a preprocessing overhead, we demonstrated how this cost can be amortized across multiple queries, which makes it an efficient and scalable solution.

6 Limitations

6.1 Approximation for Attention Vector.

Despite our intention of making the reaction vector the likelihood ratio of joint probabilities, the current attention vector calculation is not a truly joint probability but an approximation due to taking the naive *arithmetic mean* and disregarding the masked values within the self-attention matrix in Equation 5. Taking the column-wise geometric mean across multi-heads is desired but left for future work due to edge cases such as one of the attention values being zero.

6.2 Attention as Importance Probabilty

We are aware that the debate on whether attention weights do (Xiao et al., 2024; Zhang et al., 2023) or do not (Guo et al., 2024) necessarily reflect ground-truth importance. However, we find that in practice, this heuristic remains effective for retrieval. Our approach captures relative shifts in attention triggered by the query, which serves as a strong proxy for semantic relevance—something not captured by raw attention alone.

6.3 YOURA Scalability.

For each partitioned input (ensuring each token sequence fits within the context window for prefill), YOURA could retrieve in parallel. However, this optimization would require complex GPU memory management, which we leave for future work.

Despite the performance benefits, partitioning restricts holistic retrieval to the context window. Alternative windowed attention mechanisms, such as sliding or overlapping attention, could be applied; however, determining the reaction score for overlapping sentences is non-trivial and is also left for future exploration.

Acknowledgments

We extend our sincere thanks to the anonymous reviewers for their thoughtful comments that greatly improved this manuscript.

References

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. 2024. LongBench: A bilingual, multitask benchmark for long context understanding. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137, Bangkok, Thailand. Association for Computational Linguistics.

Parishad BehnamGhader, Santiago Miret, and Siva Reddy. 2022. Can retriever-augmented language models reason? the blame game between the retriever and the language model. *arXiv preprint arXiv:2212.09146*.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. 2024. Unlimiformer: Long-range transformers with unlimited length input. *Advances in Neural Information Processing Systems*, 36.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. 2022. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091.

Amnon Catav. 2023. Less is more: Why use retrieval instead of larger context windows.

Harrison Chase. 2022. Langchain.

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

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023b. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*.

Sunhao Dai, Yuqi Zhou, Liang Pang, Weihao Liu, Xiaolin Hu, Yong Liu, Xiao Zhang, Gang Wang, and Jun Xu. 2024. Neural retrievers are biased towards llm-generated content. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 526–537.

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.

Ginger Delmas, Rafael Sampaio de Rezende, Gabriela Csurka, and Diane Larlus. 2022. Artemis: Attention-based retrieval with text-explicit matching and implicit similarity. *Preprint*, arXiv:2203.08101.

Yiran Ding, Li Lyna Zhang, Chengruidong Zhang, Yuanyuan Xu, Ning Shang, Jiahang Xu, Fan Yang, and Mao Yang. 2024. Longrope: Extending llm context window beyond 2 million tokens. *arXiv preprint arXiv:2402.13753*.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2024. The faiss library. *arXiv preprint arXiv:2401.08281*.

Alexander Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, Florence, Italy. Association for Computational Linguistics.

John Giorgi, Luca Soldaini, Bo Wang, Gary Bader, Kyle Lo, Lucy Lu Wang, and Arman Cohan. 2022. Exploring the challenges of open domain multi-document summarization. *ArXiv, abs/2212.10526*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Jiafeng Guo, Yinqiong Cai, Yixing Fan, Fei Sun, Ruqing Zhang, and Xueqi Cheng. 2022. Semantic models for the first-stage retrieval: A comprehensive review. *ACM Trans. Inf. Syst.*, 40(4).

Zhiyu Guo, Hidetaka Kamigaito, and Taro Watanabe. 2024. Attention score is not all you need for token importance indicator in KV cache reduction: Value also matters. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21158–21166, Miami, Florida, USA. Association for Computational Linguistics.

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 *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. *Preprint*, arXiv:2310.06825.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023b. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *Preprint*, arXiv:2310.06839.

Zhengbao Jiang, Luyu Gao, Jun Araki, Haibo Ding, Zhiruo Wang, Jamie Callan, and Graham Neubig. 2022. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. *Preprint*, arXiv:2212.02027.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Greg Kamradt. 2024. Retrieval tutorials: Five levels of chunking strategies in rag. `https://github.com/FullStackRetrieval-com/RetrievalTutorials/tree/main/tutorials/LevelsOfTextSplitting`.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48.

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.

Bernhard Kratzwald, Anna Eigenmann, and Stefan Feuerriegel. 2019. RankQA: Neural question answering with answer re-ranking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6076–6085, Florence, Italy. Association for Computational Linguistics.

Amrith Krishna, Sebastian Riedel, and Andreas Vlachos. 2022. Proofver: Natural logic theorem proving for fact verification. *Transactions of the Association for Computational Linguistics*, 10:1013–1030.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. 2024. Babilong: Testing the limits of llms with long context reasoning-in-a-haystack. *Preprint*, arXiv:2406.10149.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Shanda Li, Chong You, Guru Guruganesh, Joshua Ainslie, Santiago Ontanon, Manzil Zaheer, Sumit Sanghai, Yiming Yang, Sanjiv Kumar, and Srinadh Bhojanapalli. 2023. Functional interpolation for relative positions improves long context transformers. *arXiv preprint arXiv:2310.04418*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, page 1–7, USA. Association for Computational Linguistics.

Jerry Liu. 2022. LlamaIndex.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Zichang Liu, Jue Wang, Tri Dao, Tianyi Zhou, Binhang Yuan, Zhao Song, Anshumali Shrivastava, Ce Zhang, Yuandong Tian, Christopher Re, et al. 2023. Deja vu: Contextual sparsity for efficient llms at inference time. In *International Conference on Machine Learning*, pages 22137–22176. PMLR.

Chao Lou, Zixia Jia, Zilong Zheng, and Kewei Tu. 2024. Sparser is faster and less is more: Efficient sparse attention for long-range transformers. *Preprint*, arXiv:2406.16747.

Kun Luo, Zheng Liu, Shitao Xiao, and Kang Liu. 2024. Bge landmark embedding: A chunking-free embedding method for retrieval augmented long-context large language models. *Preprint*, arXiv:2402.11573.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Ziming Mao, Chen Henry Wu, Ansong Ni, Yusen Zhang, Rui Zhang, Tao Yu, Budhaditya Deb, Chenguang Zhu, Ahmed Awadallah, and Dragomir Radev. 2022. DYLE: Dynamic latent extraction for abstractive long-input summarization. In *Proceedings of the*

60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1687–1698, Dublin, Ireland. Association for Computational Linguistics.

Pedro Henrique Martins, Zita Marinho, and André FT Martins. 2021. ∞-former: Infinite memory transformer. *arXiv preprint arXiv:2109.00301*.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *arXiv preprint arXiv:2305.16300*.

Gianluca Moro and Luca Ragazzi. 2023. Align-then-abstract representation learning for low-resource summarization. *Neurocomputing*, 548:126356.

Gianluca Moro, Luca Ragazzi, Lorenzo Valgimigli, Giacomo Frisoni, Claudio Sartori, and Gustavo Marfia. 2023. Efficient memory-enhanced transformer for long-document summarization in low-resource regimes. *Sensors*, 23(7).

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.

Ronak Pradeep, Sahel Sharifymoghaddam, and Jimmy Lin. 2023. Rankzephyr: Effective and robust zero-shot listwise reranking is a breeze! *Preprint*, arXiv:2312.02724.

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*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *Preprint*, arXiv:2003.07082.

Hongjin Qian, Peitian Zhang, Zheng Liu, Kelong Mao, and Zhicheng Dou. 2024. Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery. *Preprint*, arXiv:2409.05591.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Devendra Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6648–6662, Online. Association for Computational Linguistics.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. Asqa: Factoid questions meet long-form answers. *arXiv preprint arXiv:2204.06092*.

Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14918–14937, Singapore. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Boxin Wang, Wei Ping, Peng Xu, Lawrence McAfee, Zihan Liu, Mohammad Shoeybi, Yi Dong, Oleksii Kuchaiev, Bo Li, Chaowei Xiao, et al. 2023. Shall we pretrain autoregressive language models with retrieval? a comprehensive study. *arXiv preprint arXiv:2304.06762*.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. 2024. Augmenting language models with long-term memory. *Advances in Neural Information Processing Systems*, 36.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Qingyang Wu, Zhenzhong Lan, Kun Qian, Jing Gu, Alborz Geramifard, and Zhou Yu. 2020. Memformer: A memory-augmented transformer for sequence modeling. *arXiv preprint arXiv:2010.06891*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. *Preprint*, arXiv:2309.17453.

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 *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Tianzhu Ye, Li Dong, Yuqing Xia, Yutao Sun, Yi Zhu, Gao Huang, and Furu Wei. 2024. Differential transformer. *Preprint*, arXiv:2410.05258.

Tianjun Zhang, Shishir G. Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E. Gonzalez. 2024. Raft: Adapting language model to domain specific rag. *Preprint*, arXiv:2403.10131.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022. Summ^n: A multi-stage summarization framework for long input dialogues and documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1592–1604, Dublin, Ireland. Association for Computational Linguistics.

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. *Preprint*, arXiv:2306.14048.

Jiawei Zhao, Zhenyu Zhang, Beidi Chen, Zhangyang Wang, Anima Anandkumar, and Yuandong Tian. 2024. Galore: Memory-efficient llm training by gradient low-rank projection. *arXiv preprint arXiv:2403.03507*.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. QMSum: A new benchmark for query-based multi-domain meeting summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921, Online. Association for Computational Linguistics.

Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. 2021. Long-short transformer: Efficient transformers for language and vision. *Advances in neural information processing systems*, 34:17723–17736.

A Risks

Bias and Fairness Concerns. YOURA optimizes attention-based retrieval, which may harmfully be trained to inject biases during pre-training. If certain perspectives or sources receive stronger attention weights, models might over-represent dominant narratives while underexposing marginalized voices. Future work should explore bias correction techniques, including obfuscated mechanism or fairness-aware retrieval objectives, to ensure more balanced outputs.

Privacy and Security Considerations. YOURA benefits from preprocessed data caching. For example, if the sentence extraction algorithm output is improperly protected, unintentional data leakage may occur. We recommend not using YOURA for privacy-preserving mechanisms. If must, controlled access to preprocessed data and robust anonymization techniques are necessary to minimize risks.

Exclusion and Underrepresentation Risks. As with many NLP advancements, YOURA's retrieval effectiveness may be language-dependent, especially when the foundational model favors high-resource languages. Future research should explore methods to fine-tune reaction scoring across diverse linguistic datasets to ensure inclusivity and avoid exacerbating linguistic disparities.

B Scientific Artifacts

Dataset. The two main dataset used are English version of LongBench (Bai et al., 2024) (MIT License) and BABILong (Kuratov et al., 2024) (Apache 2.0 license). The LongBench is an umbrella benchmark dataset that cites NarrativeQA (Kočiský et al., 2018), Qasper (Dasigi et al., 2021), MultiFieldQA (Bai et al., 2024) (generated using arXiv, C4 dataset (Raffel et al., 2020), Wikipedia, etc.), HotpotQA (Yang et al., 2018), 2WikiMultihopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), GovReport (Huang et al., 2021), QMSum (Zhong et al., 2021), MultiNews (Fabbri et al., 2019), TREC (Li and Roth, 2002), TriviaQA (Joshi et al., 2017), SAMSum (Gliwa et al., 2019). BABILong leveraged PG19 dataset (Rae et al., 2019) as the "haystack" text.

Models. We use Llama2 (Touvron et al., 2023) (https://www.llama.com/llama2/license/), Llama3 (Grattafiori et al., 2024) (https://www.llama.com/llama3/license/), Mistralv0.2 (Jiang et al., 2023a) (Apache 2.0 license), Llama3.1 (Grattafiori et al., 2024) (https://www.llama.com/llama3_1/license/), Qwen2.5 (Qwen et al., 2025) (Apache 2.0 license) in our paper.

Some embedding models used in the paper include `paraphrase-MiniLM-L6-v2` (Apache 2.0 license), E5-Mistral (MIT License).

Software. We built upon HuggingFace transformer library (Wolf et al., 2020) (Apache 2.0 License) and vLLM (Kwon et al., 2023) (Apache 2.0 License) for our experiments.

License. License details are provided within along the artifact.

C AI Writing Assistant

Throughout the paper, writing is done by human and human used AI as writing assistant responsible for proof-reading, reviewer-like human providing feedbacks on writing structure, clarity, and conciseness. Futhermore, some math formula annotations in Section 3, such as annotation to represent column-wise subsequencing with subscript and superscript outside of a paranthesis, was AI's suggestion. AI assisted in finding proper description of the reaction score such as "likelihood of normalized joint probability" in Section 3.2, but the original idea of leveraging geomentric mean for interpreting it as a likelihood was human effort. We used Google Gemini 2.5 family for camera-ready proofreading and for all others used OpenAI ChatGPT 4 family.

D Discussion

D.1 Non-QA Accuracy.

Table 8 shows how YOURA impacts other tasks such as summarization, few-shot learning, and synthetic tasks such as counting unique passages (**PassageCount**) or finding a passage that an abstract is generated from (**PassageRetrieval**). We make the following observations: (1) YOURA effectively reduces the input size with a small impact on the summarization quality, (2) YOURA negatively impacts the few-shot learning or **PassageCount** tasks, (3) YOURA improves **PassageRetrieval** for Llama3 but not for Llama2 or Mistralv0.2.

For **Summarization** task, the LM summarizes the report, meeting notes, or news. For GovReport (3-1) and MultiNews, the LongBench does not include data-specific queries, and we use a portion of the prompt as the query. YOURA performs slightly better than the truncate-middle approach (+0.02). It indicates that YOURA can retrieve the sentences crucial for the correct summary.

For **Few-Shot Learning** task, the LM is given several exemplary QA pairs or dialogues and the real query. We observe that the truncate-middle outperformed by a huge margin and reveals that YOURA is not a silver bullet for all tasks. It is not surprising because retrieval has two potential failure points: (i) chunking may tear apart the question-answer pair or dialogue, (ii) retrieval may retrieve an incomplete example or dialogue distracting the real query intention, and (iii) retrieval may be biased towards specific type of question-answer pair. In contrast, truncate-middle may have at most two torn examples or dialogues because it splits into head, middle, and tail.

In ***PassageCount***, LM should return the unique passage count from a list of passages with duplicates. The task requires a full view of each passage. Retrieving a list of passage segments results in low accuracy, resulting in a single-digit accuracy across models and setups. YOURA performs worse than the truncate-middle approach across all models.

In ***PassageRetrieval***, the LM is given a summary and thirty passages to identify the source passage. YOURA shows better accuracy for Llama3 but worse for other models. The mixed result implies that YOURA improves accuracy when the model context window size is smaller than the input. Still, the filtered information negatively impacts accuracy when the context window size is sufficiently large (e.g., Mistralv0.2 has 32 K window size).

D.2 Models with Long Context Windows.

Models with long context window sizes such as Llama3.1 (128 K token context window size) reduce the importance of retrieval from the accuracy perspective. In such models, the whole input could be processed without truncation at the cost of additional computation and memory capacity requirements. However, the strength of YOURA lies in the fine-tuning and training-free aspect along with the benefits of the RAG systems, computational efficiency, and resource budget-friendly.

LM	Retrieval	Summ Avg	F-Shot Avg	P.C. (5-1)	P.R. (5-2)
L2	Trunc	**24.43**	**62.91**	**2.60**	**9.25**
	YOURA	24.12	13.45	2.07	3.62
L3	Trunc	26.86	**68.67**	**8.00**	67.50
	YOURA	**26.88**	30.80	5.00	**87.00**
M	Trunc	**27.48**	**67.33**	**4.30**	**88.02**
	YOURA	27.41	30.80	3.06	71.98

L2: Llama2-7B, L3: Llama3-8B, M: Mistralv0.2

Table 8: Accuracy of non-QA tasks: summarization, few-shot learning, and synthetic tasks (**P**assage **C**ount, **P**assage **R**etrieval).

Retrieval	SDoc*	MDoc	Summ	Avg
Trunc.	51.42	46.05	28.79	40.92
YOURA	47.70	42.32	28.37	38.43 (-6%)

*Excluding `1-1` due to out-of-memory issues with Trunc

Table 9: Accuracy drop of YOURA when compared to truncation for Llama 3.1. Llama 3.1's context window size is 128 K tokens, resulting in no truncation for all of the tested datasets.

D.3 Qwen 2.5 7B Instruct Result

Table 10 shows detailed result of Qwen2.5-7B-Instruct-32K. We observed that some attention heads produced `NaN` (Not a Number) scores, which made the standard YOURA algorithm unstable. To address this, we present three variants of YOURA, each handling the `NaN` values differently.

- YOURA-mean: This variant uses the standard torch.mean function, which propagates `NaN` values. As a result, contexts producing `NaN` scores are excluded from the evaluation by naively truncating from the beginning to fit within the context window size.
- YOURA-nanmean: This variant replaces the standard mean with a `NaN`-aware mean function (e.g., torch.nanmean), which ignores `NaN` values during the calculation.
- YOURA-zero: In this variant, `NaN` values are explicitly replaced with 0 before the mean is computed.

D.4 Alternative Reaction Score.

To show that our definition of reaction score works better than others we compare an alternative definition, the absolute difference between initial and

LM	Retrieval	Single-Document QA 1-1	1-2	1-3	SDoc Avg	Multi-Document QA 2-1	2-2	2-3	MDoc Avg	Overall Avg
Qwen 2.5	No Ctxt	7.06	11.00	15.69	11.25	27.29	26.12	11.90	21.77	16.51
	B500_63	16.37	40.43	45.56	34.12	49.66	40.44	18.34	36.15	35.13
	B50_630	22.69	40.92	50.64	38.08	53.70	45.71	27.98	42.46	40.27
	E500_63	16.37	40.43	45.56	34.12	49.66	40.44	18.34	36.15	35.13
	E50_630	24.16	40.28	48.22	37.55	56.34	42.96	34.08	44.46	41.01
	YOURA - mean	24.21	**43.84**	47.12	38.39	53.79	41.01	29.12	41.31	39.85
	YOURA - nanmean	**31.55**	43.53	**50.95**	**42.01**	**57.06**	46.02	28.93	44.00	43.01
	YOURA - zero	30.57	43.17	50.92	41.55	57.04	**47.02**	**29.99**	**44.69**	**43.12**

The **bold score** indicates the best score for each pre-trained model.

Table 10: Detailed experiment results for Qwen 2.5.

Reaction Vector Definition	SDoc	MDoc	Avg
abs(Reacted - $\lambda\times$ Initial)			
$\lambda = 1$	36.57	36.44	36.51
$\lambda = 0.75$	37.75	37.85	37.80
$\lambda = 0.5$	37.98	**38.80**	38.39
$\lambda = 0.25$	37.52	38.49	38.01
Reacted / Initial	**38.82**	38.44	**38.63**

Table 11: Comparison against the alternative reaction score definitions (Model: LLlama3-8B).

reacted attention vectors. We study how the change of relative ration between the Reacted and Initial attention impact the QA task accuracy. Table 11 shows that among the several potential alternative definitions, ratio performs the best.

E Sentence Extraction Algorithm

Algorithm 1 lists the details of the sentence extraction algorithm.

F Dataset

Figure 12 lists dataset details. For BABILong dataset, we used pregenerated RMT-team/babilong from huggingface repository's 8K and 32K subsets.

G Detailed Results

LongBench QA Evaluation Details. Table 13, Table 14 are the detailed version of Table 2 and Table 4, respectively.

Sentence Extraction Accuracy. Table 15 and Table 16 shows full result of sentence extraction algorithm's accuracy and quality in Table 6.

Algorithm 1 Sentence Extraction Algorithm

Require: seq - Encoded token sequence
Require: TS - Target Sentences generated by NLP models
Require: $tol \leftarrow 30$ - Max adjustment attempts
$P \leftarrow []$ {Processed sentences}
$B \leftarrow []$ {Sentence boundary indices}
$i, m \leftarrow 0$ {Initialize indices}
while $P \neq TS$ **do**
  APPEND $TS[i]$ to P
  $c \leftarrow$ Length(Encode(Concat(P))) {Candidate}
  $saved \leftarrow c$
  $visited \leftarrow \emptyset$
  while true **do**
    if $c \in visited$ or $c > |seq|$ or $|c - saved| > tol$ **then**
      $c \leftarrow saved$
      break
    end if
    ADD c to $visited$
    $s \leftarrow$ Decode($seq[m : c]$)
    if $s == TS[i]$ **then**
      break
    else if s is substring of $TS[i]$ **then**
      $c \leftarrow c + 1$
    else
      $c \leftarrow c - 1$
    end if
  end while
  APPEND c to B
  $m \leftarrow c$
  $i \leftarrow i + 1$
end while
return B

Dataset	ID	Source	Avg # Tokens Llama2	Llama3	Mistral	Metric	# Data
LongBench - Single-Document QA							
NarrativeQA	1-1	Literature, Film	35937	29777	35937	F1	200
Qasper	1-2	Science	5603	4922	5517	F1	200
MultiFieldQA-en	1-3	Multi-field	8046	6889	7877	F1	150
LongBench - Multi-Document QA							
HotpotQA	2-1	Wikipedia	15244	12780	14890	F1	200
2WikiMultihopQA	2-2	Wikipedia	8381	7096	8281	F1	200
MuSiQue	2-3	Wikipedia	18459	15544	18069	F1	200
LongBench - Summarization							
GovReport	3-1	Government report	12235	10240	11631	Rouge-L	200
QMSum	3-2	Meeting	15907	13853	15791	Rouge-L	200
MultiNews	3-3	News	3113	2608	3005	Rouge-L	200
LongBench - Few-shot Learning							
TREC	4-1	Web question	7759	6755	7559	Accuracy (CLS)	200
TriviaQA	4-2	Wikipedia, Web	13212	11038	12938	F1	200
SAMSum	4-3	Dialogue	10964	8999	10684	Rouge-L	200
LongBench - Synthetic Task							
PassageCount	5-1	Wikipedia	17209	14879	16797	Accuracy (EM)	200
PassageRetrieval	5-2	Wikipedia	14228	12283	13894	Accuracy (EM)	200
BABILong - Needle-in-a-haystack							
Single Supporting Fact	QA1	PG19 as Noise	Varies	Varies	Varies	Accuracy (EM)	100
Two Supporting Facts	QA2	PG19 as Noise	Varies	Varies	Varies	Accuracy (EM)	100
Three Supporting Facts	QA3	PG19 as Noise	Varies	Varies	Varies	Accuracy (EM)	100
Two Arg Relations	QA4	PG19 as Noise	Varies	Varies	Varies	Accuracy (EM)	100
Three Arg Relations	QA5	PG19 as Noise	Varies	Varies	Varies	Accuracy (EM)	100

Table 12: Evaluation detail on LongBench dataset (Bai et al., 2024) and BABILong (Kuratov et al., 2024).

Resilience to Edge Cases. To quantify the resilience despite typos and overly representative dictionaries, we measured the Levenshtein distance between the target sentence and the sentence yielded by our sentence extraction algorithm (Table 16). On average, one needs less than 3 character edits (insert, remove, replace) for the algorithm output sentences to match the target sentences. The NZ column in Table 16 is the average Levenshtein distance of unmatched sentences. From these rows, we conclude that the sentence extraction algorithm is resilient to incorrect split and its cascading effect.

H Addtional Related Works

Long-context Language Models. Training LLMs on sequences with a maximum length while still ensuring they infer well on sequences longer than the maximum is challenging. Previous studies proposed techniques such as positional interpolation (Chen et al., 2023a; Li et al., 2023), positional extrapolation (Press et al., 2021), external emory (Wu et al., 2020; Martins et al., 2021), memory-retrieval augmentation strategies (Mohtashami and Jaggi, 2023; Wang et al., 2024) to push the limit of the context window size to 2 million and even longer (Ding et al., 2024; Bertsch et al., 2024). Although these works improved model accuracy on various tasks, the large context window size implies more computational and memory costs.

Attention Mechanism. Researchers proposed self-attention alternatives to mitigate its quadratic complexity, which becomes a computational bottleneck for a long context. These include sparse attention mechanisms with pre-defined sparsity patterns (Zhu et al., 2021; Liu et al., 2023), recurrence-based method (Bulatov et al., 2022), low-rank pro-

LM	Retrieval	Retr. Rate Avg	Single-Document QA 1-1	1-2	1-3	SDoc Avg	Multi-Document QA 2-1	2-2	2-3	MDoc Avg	Overall Avg
Llama2	N/A	INF	8.89	13.87	18.19	13.65	23.05	25.67	9.17	19.30	16.47
	B500_7	3.28	12.59	21.44	34.42	22.82	25.92	27.47	7.11	20.17	21.49
	B50_70	3.28	16.37	21.51	33.71	23.86	27.48	28.62	13.64	23.25	23.56
	E500_7	3.28	12.85	16.08	22.47	17.13	25.51	24.84	8.35	19.57	18.35
	E50_70	3.28	17.15	20.12	35.18	24.15	28.80	27.15	12.66	22.87	23.51
	Trunc.	3.28	18.84	21.71	**36.68**	25.74	27.55	31.08	8.40	22.34	24.04
	LLL	3.79	15.36	15.68	32.93	21.32	**38.99**	30.91	**21.74**	21.12	21.97
	YOURA	3.39	**19.70**	**26.97**	33.99	**26.88**	32.35	**32.57**	14.02	**26.31**	**26.59**
Llama3	N/A	INF	9.62	14.38	10.78	11.59	28.43	30.46	10.60	23.16	17.37
	B500_15	1.57	18.27	40.21	43.72	34.07	41.75	41.18	16.88	33.27	33.73
	B50_150	1.57	19.86	36.51	42.39	32.92	47.04	40.58	19.86	35.83	34.17
	E500_15	1.57	16.38	39.85	43.76	33.33	42.28	34.93	15.25	30.82	32.08
	E50_150	1.57	19.05	35.44	40.02	31.50	48.05	36.85	22.19	35.70	33.60
	Trunc.	1.57	21.54	44.21	44.75	36.83	46.43	36.23	21.50	34.72	35.93
	LLL	1.94	20.95	36.35	41.86	33.05	47.53	**41.03**	**27.20**	**38.59**	35.82
	YOURA	1.76	**26.32**	**45.22**	**44.91**	**38.82**	**50.51**	37.96	26.87	38.44	**38.63**
Mistralv0.2	N/A	INF	7.24	7.69	6.13	7.02	16.41	11.87	6.13	11.47	9.25
	B500_63	1.04	20.74	29.22	46.91	32.29	34.26	22.26	17.47	24.66	29.02
	B50_630	1.04	18.84	25.08	43.50	29.14	32.85	18.66	18.32	23.28	26.63
	E500_63	1.04	14.02	24.14	42.22	26.79	32.81	17.55	15.82	22.06	24.76
	E50_630	1.04	18.84	25.08	43.50	29.14	32.85	18.66	18.32	23.28	26.63
	Trunc.	1.04	20.80	29.23	**47.92**	32.65	**37.28**	21.72	20.86	26.62	**29.64**
	LLL	1.08	18.85*	22.89	42.23	27.99	36.56	**23.88***	**22.93***	**27.79**	27.89
	YOURA	1.33	**22.57**	**30.07**	46.86	**33.22**	36.63	22.06	19.48	26.06	**29.64**

The **bold score** indicates the best score for each pre-trained model.

Table 13: Detailed experiment results for LongBench's single-document and multi-document QA datasets. YOURA shows better quality with a higher retrieval ratio — i.e., retrieved fewer tokens. All retrieval scenarios use the maximal budget (4K, 8K, and 32K tokens depending on model) except for no-context setup and "YOURA". Retrieval ratio (second column) is an average of the total token count divided by the retrieved token count. Mistralv0.2 with LongLLMLingua retrieval on 17, 2, and 9 data points from 1-1 , 2-2 , 2-3 dataset, respectively, resulted in runtime error and we report the average of the remaining results.

jection attention (Zhao et al., 2024), and memory-based mechanisms (Lou et al., 2024). Leveraging attention for retrieval has been explored in image retrieval (Delmas et al., 2022) or QA tasks (Jiang et al., 2022), learning the inferred relevance from attention for retrieval. However, these approximate methods introduce inductive bias (e.g., predefined sparsity) that can fit well for specific domains, but may reduce model quality in general LLM training.

Differential Transformer (Ye et al., 2024) is the concurrent work that is closely related to our work, but differs in the following ways: (1) DIFF proposed new differential transformer architecture requiring a full model training, while our proposal works on off-the-shelf language models without training or fine-tuning, (2) DIFF focuses on reducing the hallucinations by reducing the attention to irrelevant tokens while our proposal focuses on retrieving the relevant sentences.

Fine-tuning for Long Context. Recent works (Ding et al., 2024; Chen et al., 2023b; Peng et al., 2023) show that a pre-trained LLM context window can be extended by fine-tuning on longer texts. However, fine-tuning approaches still require multiple self-attention computations demanding both high-quality training datasets and computational costs. Our approach does not require fine-tuning.

Summarization Recent advances in the field of long-input summarization have led to a variety of innovative approaches addressing both general and domain-specific challenges, particularly under low-resource conditions (Zhang et al., 2022; Moro

Model	Retrieval Method	Single-Document QA 1-1	1-2	1-3	Multi-Document QA 2-1	2-2	2-3	Overall Avg Req./Sec
Llama2	Trunc.	5.58	4.68	5.50	**5.71**	5.87	**5.68**	5.50
Llama2	YOURA	**5.70**	**4.88**	**5.72**	5.70	**6.00**	5.63	**5.60 (+2%)**
Llama3	Trunc.	2.50	3.95	3.23	2.63	3.23	2.52	3.01
Llama3	YOURA	**2.60**	**4.87**	**3.82**	**2.72**	**3.83**	**2.53**	**3.39 (+13%)**
Mistralv0.2	Trunc.	0.55	3.17	2.19	1.08	2.14	0.86	1.66
Mistralv0.2	YOURA	**0.64**	**4.02**	**2.97**	**1.45**	**2.85**	**1.15**	**2.18 (+31%)**

Table 14: vLLM Serving performance (request per second, higher the better) for LongBench's single-document and multi-document QA datasets with respect to Truncation and YOURA. YOURA retrieved fewer tokens resulting in upto 30% higher inference throughput. We report single run due to restricted computing resource.

LM	1-1	1-2	1-3	2-1	2-2	2-3	Avg
Average Sentence Exact Match ↑							
L2	0.941	0.831	0.935	0.984	0.988	0.978	0.943
L3	0.999	0.996	0.965	0.865	0.869	0.856	0.925
M	0.940	0.831	0.935	0.984	0.988	0.978	0.943
Avg	0.960	0.886	0.945	0.944	0.948	0.937	0.937

L2: Llama2-7B, L3: Llama3-8B, M: Mistralv0.2

Table 15: Sentence extraction algorithm's accuracy measured as the average sentence EM rate (exact match / total # of sentences)

LM	1-1	1-2	1-3	2-1	2-2	2-3	Avg
Average of Edit Distances↓							
L2	0.01	0.03	0.22	0.02	2.08	0.77	0.52
L3	0.33	0.36	0.18	0.37	0.01	0.18	0.24
M	0.01	0.03	0.23	0.02	2.15	0.77	0.54
Average of Non-Zero Edit Distances↓							
L2	0.78	1.55	1.82	1.40	2.89	4.08	2.09
L3	2.52	2.61	1.29	2.69	1.11	1.35	1.93
M	0.81	1.58	1.84	1.42	2.96	4.10	2.12

L2: Llama2-7B, L3: Llama3-8B, M: Mistralv0.2

Table 16: Resilience of Sentence Extraction algorithms measured as the average edit distance between the target sentence and algorithm output. A smaller number indicates fewer character edits (insert, remove, replace) to transform the TLSC output sentence to the target sentence, and thus a more resilient quality.

and Ragazzi, 2023; Moro et al., 2023; Mao et al., 2022; Liu et al., 2024). These contributions emphasizes how summarization could mitigate the long-document challenge with limited context window LMs.

Reranking & Multi Stage Retrieval Reranking (Kratzwald et al., 2019; Sun et al., 2023; Pradeep et al., 2023) and multi stage retrieval is different from the first stage retriever because it further filters information from an already retrieved contexts as the second phase of retrieval (Guo et al., 2022). Comparing these works against our work is beyond the scope.

I YOURA Latency Breakdown

Figure 3 illustrates the overhead of naive YOURA implementation and how reusing KVCache reduces the overhead. The left figure is the latency breakdown for a single QA execution and the right figure is the per processed token latency breakdown (2× # of input tokens for retrieval, # of retrieved tokens for generation).

From the left figure, we observe that YOURA's retrieval could be up to $5.5\times$ longer than the generation stage but performance optimization reduces the time by half. The culprit behind the overhead is that every token must be prefilled at least once, whereas during the generation stage, at most the context window-sized tokens are prefilled.

Dividing the time by the number of processed tokens reveals that the YOURA's retrieval overhead per token is smaller than the generation stage (right figure). Similar to the aggregated retrieval time, applying the performance optimization reduces the per-token latency by roughly half. Generation may take longer for each token (as observed in Llama2

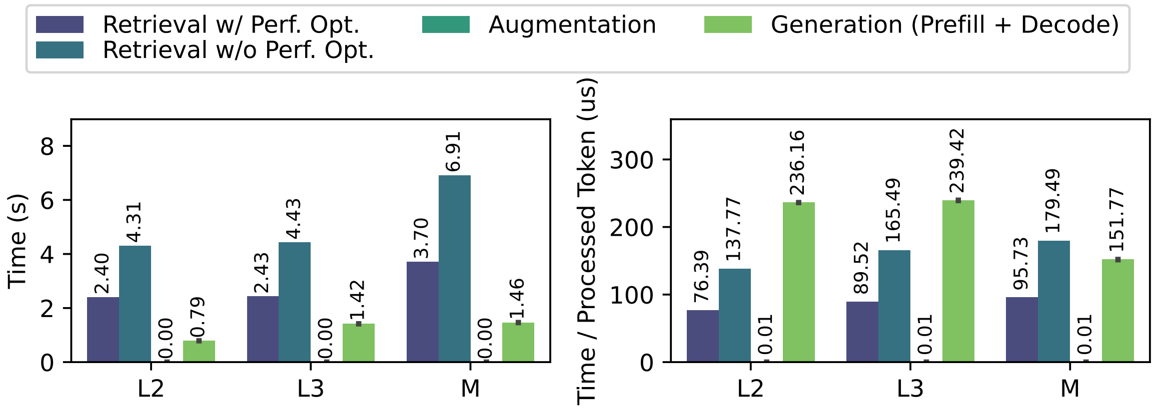

Figure 3: Latency breakdown of YOURA for each of Retrieve, Augment, Generate stage for six QA datasets. The left figure is the overall time, and the right is the amortized time per processed token. For the retrieval stage, we show both the unoptimized performance and the optimized implementation. Augmentation overhead is negligible.

and Llama3) because generation time includes both the prefill and decode stages contributing to non-linear scaling with respect to the number of processed tokens.

In the long run, YOURA's latency can be improved by reusing initial self-attention across multiple queries for the same context. This reuse not only reduces computational overhead but also boosts throughput by minimizing redundant processing. Additionally, parallel processing of partitioned input sequences can further enhance efficiency, allowing YOURA to better handle large contexts at scale.

J Config

LongLLMLingua We used the following parameters for LongLLMLingua's compression function (`compress_prompt`) (Jiang et al., 2023b). We installed using `pip install llmlingua` (version 0.2.2).

- `context`: Raw input text
- `instruction`: Prepending prompt (i.e., prompt string before the beginning of the context).
- `question`: Query
- `condition_in_question`: "after_condition"
- `reorder_context`: "sort"
- `condition_compare`: True
- `rank_method`: "longllmlingua"
- `rate`: -1, the value may be ignored by the target_tokens.
- `target_token`: 3500, 7500, and 31500, for Llama2, Llama3, and Mistralv0.2 respectively. Note that the actual retrieval is at most target tokens and impacts the retrieval process.

Mistralv0.2 For 28 out of 1150 QA task data, LongLLMLingua encounters runtime error similar to the closed issue on github repository[2].

Reaction Triggering Query For some data in summarization or few-shot task, explicit query is not provided but rather the prompt describes the task. For such case, we explicitly use the following to compute the YOURA's reaction vector:

- `gov_report`: "You are given a report by a government agency. Write a one-page summary of the report.."
- `multi_news`: "You are given several news passages. Write a one-page summary of all news."
- `passage_count`: "Enter the final count of unique paragraphs after removing duplicates. The output format should only contain the number, such as 1, 2, 3, and so on.\n\nThe final answer is: "

Rogue
- Version: 1.0.1
- Function: `get_score`
- Parameter: `avg=True`
- Reports: `scores["rouge-1"]["f"]`

Stanza
- Version: 1.8.1
- Language: `"en"`
- Processors: `"tokenize,mwt,ner"`

[2]https://github.com/microsoft/LLMLingua/issues/14