QuickLLaMA: Query-aware Inference Acceleration for Large Language Models

Jingyao Li[1], **Han Shi**[2], **Sitong Wu**[1], **Chuanyang Zheng**[1], **Pengguang Chen**[3], **Zhenguo Li**[2], **Xin Jiang**[2], **Hong Xu**[1], **Jiaya Jia**[1,3]

[1]CUHK, [2]Huawei Noah's Ark Lab, [3]SmartMore

Abstract

The capacity of Large Language Models (LLMs) to comprehend and reason over long contexts is pivotal for advancements in diverse fields. Yet, they still struggle with identifying relevant contexts and memory searching. To address this issue, we introduce Query-aware Inference for LLMs (QLLM), a system designed to process extensive sequences akin to human cognition. By focusing on memory data relevant to a given query, QLLM accurately captures pertinent information within a fixed window size and provides precise answers to queries. It requires no additional training and can be seamlessly integrated with any LLMs. Using LLaMA3 (QuickLLaMA), QLLM can read *Harry Potter* within 30 seconds and accurately answer related questions. On widely recognized benchmarks, QLLM improved performance by 7.17% compared to the current SOTA on LLaMA3 and by 3.26% on Mistral on the ∞-bench. In the Needle-in-a-Haystack and BABILong task, QLLM improved upon the current SOTA by 7.0% and 6.1%. Our code is in https://github.com/dvlab-research/Q-LLM.

1 Introduction

The ability to understand and reason over broad contexts has always been a long-term research focus of Large Language Models (LLMs) (Dong et al., 2023). LLM-driven agents need to process ongoing information from external sources, which requires a strong ability to manage lengthy sequences (Li et al., 2024b; Zheng et al., 2024); An ideal ChatBot assistant should be able to operate consistently over the content of conversations spanning recent days (OpenAI et al., 2024). Other tasks such as summarizing and answering questions based on books, reports, and documents, as well as generating code at the repository level, also require the capability to handle long context sequences (Bai et al., 2023; Zhang et al., 2024).

Yet, due to the challenges posed by unobserved extensive inputs (Lin et al., 2024) and distracting, noisy contexts (Liu et al., 2023; Tworkowski et al., 2024a), most LLMs that are pre-trained on sequences comprising a few thousand tokens struggle to generalize on longer sequences, resulting in unsatisfactory performance (Press et al., 2022; Zhao et al., 2023). Some contemporary studies make use of sliding windows to disregard distant contexts, thereby ensuring that the length of the sequence do not surpass the LLMs' maximum capacity (Xiao et al., 2024b; Lin et al., 2024) and incorporate block-level context memory, which opts pertinent information from memory to disregard irrelevant disturbances (Xiao et al., 2024a). However, the memory to be focused on should differ according to the specific query requirements. Yet, for distinct queries, InfLLM (Xiao et al., 2024a) exhibits identical focal points when reading the long context, as shown in Fig. 2.

To address these challenges, we design Query-aware Inference for LLMs (QLLM), which processes extensive sequences in a manner similar to human cognition. Humans, when interpreting text, initially examine the question, and then seek the answer within the context, keeping the query in mind. This idea forms the foundation of our Query-aware Context Lookup strategy. Only memory data pertinent to the query is chosen for each computational step, disregarding unrelated distractions. As a result, LLMs can capture pertinent information within a fixed window size and provide precise answers to queries. QLLM doesn't require extra training and can be seamlessly integrated with any LLMs.

We assess the performance of QLLM by utilizing LLaMA3-8B-inst (AI@Meta, 2024) (QuickLLaMA) and Mistral-7B-inst-v0.2 (Jiang et al., 2023) as foundational models. These base models are pre-trained on sequences that do not exceed 8K tokens. Instead, our QuickLLaMA can read

Proceedings of the 31st International Conference on Computational Linguistics, pages 508–528
January 19–24, 2025.

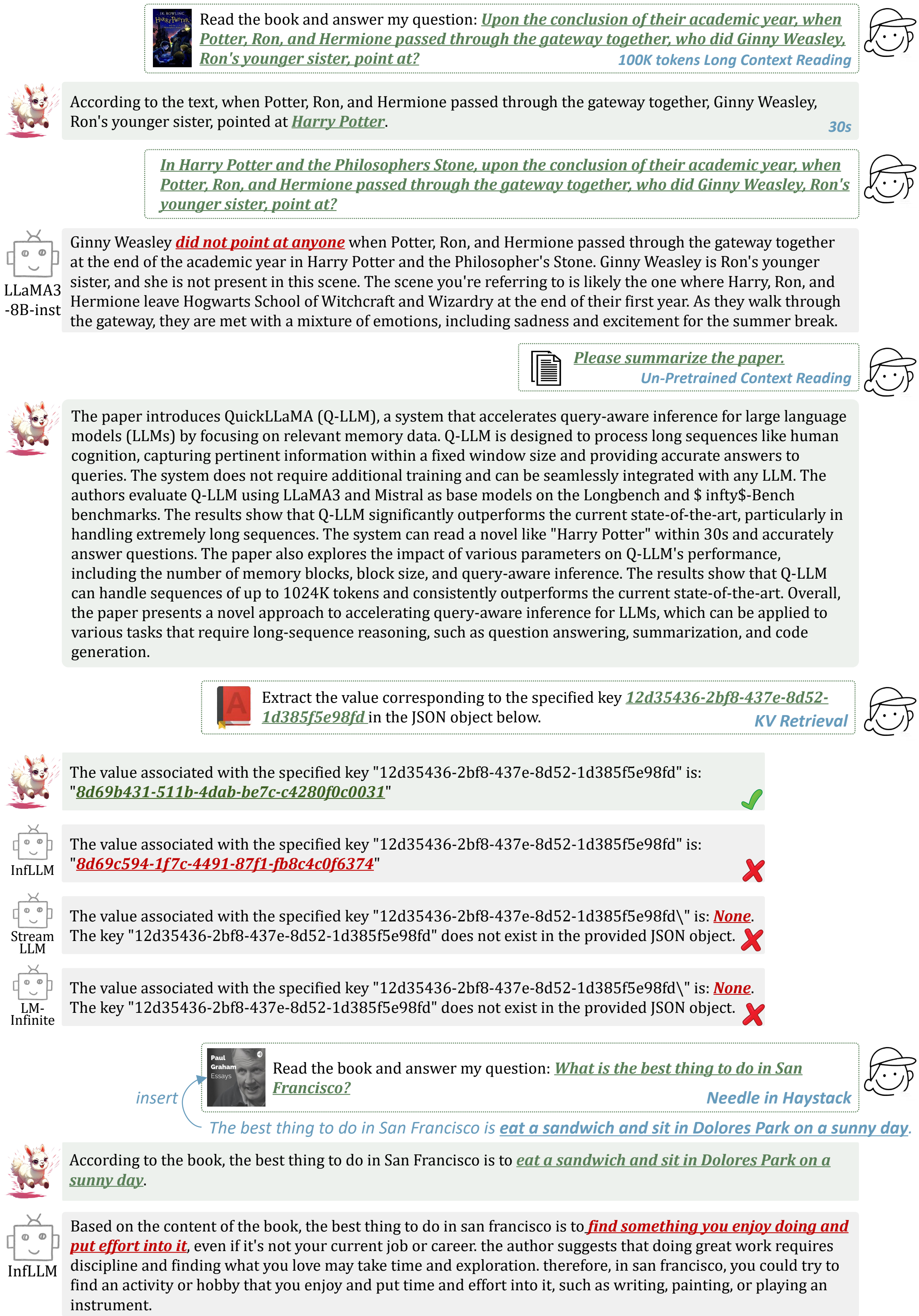


Figure 1: Examples of our QuickLLaMA-8B (1) reading long context containing 100K tokens, (2) reading our paper that has not be seen in the pretrained dataset, (3) retrieving value in long key-value pairs and (4) retrieving in Needle-in-a-Haystack task. More examples and comparisons with the SOTAs are provided in Appendix D.

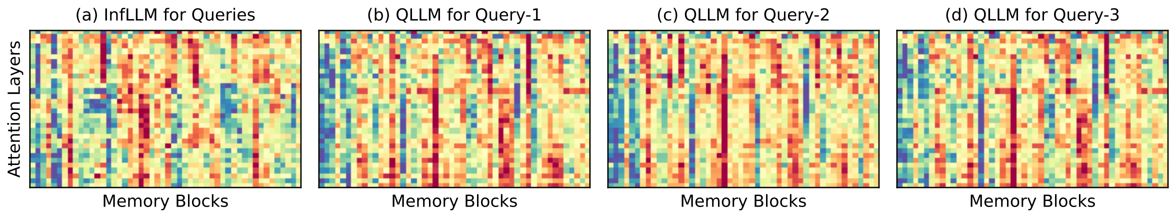


Figure 2: This is an example from the ∞-Bench. Three questions were posed about the same long book: (1) *Which among Annalisa, Seb, Peyton, and Gannonmarie is not Mrs. Bronwyn's child?* (2) *What's the name of the Bronwyns' summer home?* (3) *Who among Mrs. Bronwyn, Mrs. Deandra, Rosemarie, and Cael is the final to perish?* We present the score heatmap of the first 50 memory blocks. The methods used include (a) the consistent results from InfLLM for all three queries, and (b-d) the query-aware results from QLLM.

Harry Potter with 100K tokens within half a minute on a single A800 GPU and accurately answer the questions, as shown in Fig. 1. We employ several widely recognized benchmarks, namely Longbench (Bai et al., 2023), ∞-Bench (Zhang et al., 2024), Needle-in-a-Haystack and BABILong (Kuratov et al., 2024) Benchmark. Specifically, with a context window of 512, QLLM improved by 7.17% compared to the current SOTA on LLaMA3, and by 3.26% on Mistral on the ∞-bench. In the Needle-in-a-Haystack task, QLLM improved upon the current SOTA by 7.0% on Mistral and achieves 100% on LLaMA3. In the BABILong task, QLLM improved upon the current SOTA by 6.1%. We have extended the input sequence to contexts of 1048K length, further demonstrating our model's capability in handling extremely long sequences.

2 Related Works

Efficient Context Computation. The computational and memory demands of LLM training often limit it to short sequences. Using LLMs directly on long sequences presents challenges such as out-of-domain issues and distractions from lengthy and noisy inputs (Lin et al., 2024; Tworkowski et al., 2024a; Li et al., 2024d). As a result, context length extrapolation has emerged as a method to extend LLMs' sequence length without additional training. Early approaches have designed new relative positional encoding mechanisms during pre-training (Press et al., 2022; Tworkowski et al., 2024b). The following research has focused on the extensively adopted rotary position embedding (RoPE) (Su et al., 2023), suggesting extending the length by interpolating positions to introduce non-integer positions (Chen et al., 2024; Peng et al., 2023; Jin et al., 2024; Chen et al., 2023). To process extremely long sequences, Stream-LLM (Xiao et al., 2024b) and LM-Infinite(Lin et al., 2024) utilize the sliding window attention mechanism and discard distant contexts. Additionally, InfLLM (Xiao et al., 2024a) leverages a context memory to furnish LLMs with pertinent contextual information. Yet, the objective of these models during long-text reading is inherently ambiguous, and it can become distracting when reading extensive articles. In this work, we introduce the Query-aware Context Lookup mechanism, enabling the model to effectively retrieve information relevant to the query from lengthy texts.

Context Length Extrapolation. The computational complexity of attention layers, which grows quadratically, is a significant bottleneck restricting LLMs' capability to handle lengthy sequences. Consequently, numerous researchers have devised efficient attention mechanisms, including sparse attention (Zaheer et al., 2021; Beltagy et al., 2020; Child et al., 2019; Ainslie et al., 2020; Zhao et al., 2019), approximate attention computations using kernel functions (Kitaev et al., 2020; Wang et al., 2020; Katharopoulos et al., 2020), and replacing attention layers with state-space models of linear complexity (Gu et al., 2022; Gu and Dao, 2023). These approaches necessitate architectural modifications, requiring retraining of the models. Concurrently, many scholars have tackled this challenge from an infrastructural angle by optimizing the memory usage of attention computations to mitigate the computational resource requirements of the model (Dao et al., 2022; Dao, 2023; Hong et al., 2024; Shazeer, 2019; Kwon et al., 2023). Given the training-free nature of our method, it can be seamlessly integrated to further expedite LLM inference.

Memory-based Approaches. Memory networks have been extensively researched for decades and have demonstrated effectiveness in enhancing models with additional information storage capabil-

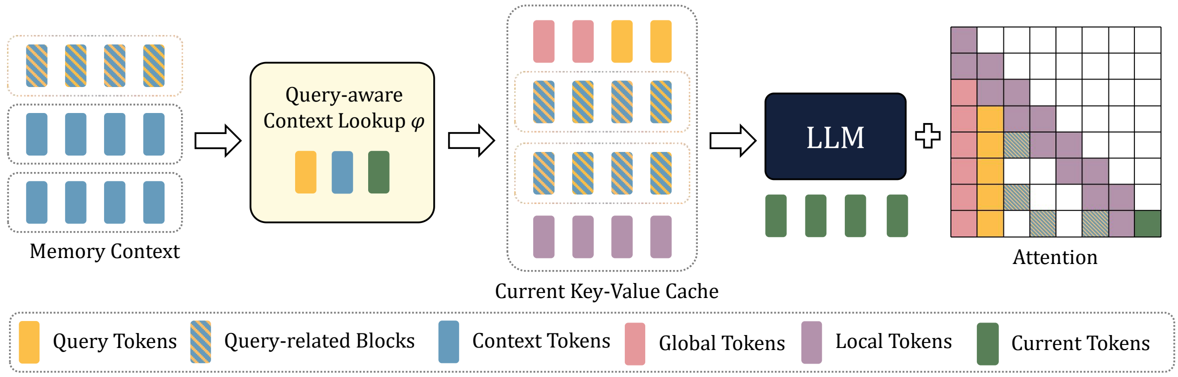


Figure 3: The illustration of our QLLM framework. The input from the memory context is partitioned into memory blocks, which are searched by Query-aware Context Lookup for query-related blocks. The current key-value cache comprises global tokens, query tokens, query-related blocks, and local tokens. Together, these form a new context window that, along with current tokens, is fed into the LLM.

ities (Graves et al., 2014; Weston et al., 2015; Sukhbaatar et al., 2015; Miller et al., 2016). With the rise of pre-trained models, memory layers have gradually found application in the training stage of recurrent transformer layers, enabling models to recursively process long sequences (Dai et al., 2019; Rae et al., 2020; Khandelwal et al., 2020; Wu et al., 2022; Bertsch et al., 2023). These approaches segment sequences, encoding each segments individually, and utilize memory to retain context information from preceding segments. Yet, they necessitate architectural modifications and are typically incorporated during the pre-training phase. In contrast, our objective is to explore the intrinsic properties of LLMs and introduce a training-free Query-aware Context Lookup mechanism for long-text comprehension.

3 Methods

In this section, we introduce the overall framework of Query-aware Inference for LLMs (QLLM) in Sec. 3.1, as depicted in Fig. 3. Then, we introduce the preliminary memory block in Sec. 3.2 and our proposed Query-aware Context Lookup in Sec. 3.3.

3.1 Overall Framework

The primary challenges in expanding the context window of LLMs arise from issues related to out-of-domain and distractions, which are a result of the extensive and noisy context. To tackle these challenges, we follow prior studies, which implement the sliding window attention mechanism (Xiao et al., 2024b; Lin et al., 2024) and the context memory module (Xiao et al., 2024a). Additionally, we propose the Query-aware Context Lookup strategy to find the query-related tokens from the context token. The past key-value vectors $\mathbf{P} = \{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^{l_P}$ consist of four composers:

1. Global tokens $\mathbf{G}$, including system prompts and task description, etc.
2. Query tokens $\mathbf{Q}$, the query of the user.
3. Context tokens $\mathbf{C}$, the context stored in the context memory, consisting of multiple memory blocks.
4. Local tokens $\mathbf{L}$, the nearest tokens to the current token.

An example of these tokens in the prompt is shown in Fig. 1. Given that all memory blocks are necessary to be maintained and most of them and seldom used, we adopt an offloading strategy, which stores most memory blocks in CPU memory. More details are in Appendix A.3. We propose the Query-aware Context Lookup strategy to find the query-related tokens $\mathbf{R}$ from the context tokens $\mathbf{C}$:

$$\mathbf{R} = \phi(\mathbf{H}, \mathbf{C}, \mathbf{Q}), \tag{1}$$

where $\phi(\cdot)$ refers to the lookup operation of context memory. We will detail the strategy in Sec. 3.3. For each step, QLLM combines the global tokens, query tokens, query-related tokens, and local tokens to compose the current key-value cache.

$$\mathbf{M} = \text{Concat}(\mathbf{G}, \mathbf{Q}, \mathbf{R}, \mathbf{L}), \tag{2}$$

Finally, the input parameters of the attention are:

$$\begin{aligned} \mathbf{A_q} &= \mathbf{P_q}\mathbf{H}, \\ \mathbf{A_k} &= \text{Concat}(\mathbf{M_k}, \mathbf{P_k}\mathbf{H}), \\ \mathbf{A_v} &= \text{Concat}(\mathbf{M_v}, \mathbf{P_v}\mathbf{H}), \end{aligned} \tag{3}$$

where $\mathbf{P_q}$, $\mathbf{P_k}$, and $\mathbf{P_v}$ are parameters in attention layers, $\mathbf{M_k}$ and $\mathbf{M_v}$ refer to the key and value vectors in the current key-value cache $\mathbf{M}$.

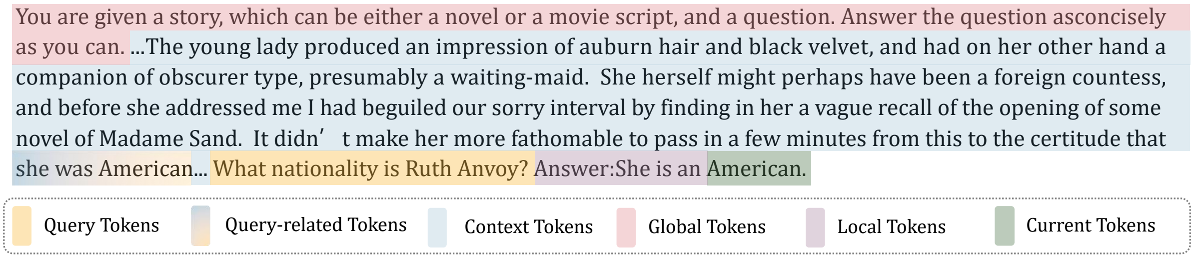


Figure 4: An example from Long-Bench. Global tokens include system prompts and task description. Query tokens represent the query of the user. Context tokens indicate the context stored in the context memory. We search query-related tokens from them, local tokens are the nearest tokens to the current token.

3.2 Memory Block

In light of the local semantic coherence in extended sequences, referring to previous studies (Xiao et al., 2024a), we perform a memory lookup at the block level. We segment the context tokens $\mathbf{C}$ into multiple memory blocks, each containing l_b tokens. We then select n_r tokens that have the highest representative scores to represent the block. For the i-th token, the representative score is calculated as

$$r_i = \frac{1}{l_L}\sum_{j=1}^{l_L} \mathbf{q}_{i+j} \cdot \mathbf{k}_i, \tag{4}$$

where l_L is the length of local token, $\mathbf{q}_{i+j}$ is the query vector for the $(i+j)$-th token and $\mathbf{k}_i$ is the key vector for the i-th token. The score r_i intuitively measures the importance of the i-th token within its local window, demonstrating its influence on other tokens in the same window.

3.3 Query-aware Context Lookup

When humans read and comprehend text, they first read the question and then search for the answer within the context with the question in mind. For instance, in Fig. 4, when reading a novel with the question *"What nationality is Ruth Anvory?"*, we can quickly locate the query-related memory context, which is *"... that she was American"*. Building on this concept, we introduce Query-aware Context Lookup, a simple but efficient lookup strategy. Our defined criterion for search is to locate tokens relevant to the query. We propose the relevance score between a memory block $\mathbf{B}$ and query tokens $\mathbf{Q}$ as follows:

$$s(\mathbf{B}, \mathbf{Q}) = \sum_{i=1}^{l_Q}\sum_{j=1}^{r_k} \mathbf{Q}_{\mathbf{q}_i} \cdot \mathbf{B}_{\mathbf{k}_j^r}, \tag{5}$$

where l_Q is the length of query tokens. $\mathbf{Q}_{q_i}$ is the i-th query vector of $\mathbf{Q}$ and $\mathbf{B}_{\mathbf{k}_j^r}$ is the j-th representative key vector of $\mathbf{B}$. The score $s(\mathbf{B}, \mathbf{Q})$ is independent of the current token $\mathbf{H}$, therefore, it only needs to be calculated once during inference.

On the other hand, the importance of different memory blocks is influenced by varying current tokens (Xiao et al., 2024a). Therefore, the relevance score with the current token is also a criterion for selecting a memory block. The relevance score between a memory block $\mathbf{B}$ and current tokens $\mathbf{H}$ is defined as:

$$s(\mathbf{B}, \mathbf{H}) = \sum_{i=1}^{l_H}\sum_{j=1}^{r_k} \mathbf{H}_{\mathbf{q}_i} \cdot \mathbf{B}_{\mathbf{k}_j^r}, \tag{6}$$

where l_H is the length of current tokens. $\mathbf{H}_{q_i}$ is the i-th query vector of $\mathbf{H}$ and $\mathbf{B}_{\mathbf{k}_j^r}$ is the j-th representative key vector of $\mathbf{B}$. The final memory block score is thus composed of these two components:

$$s(\mathbf{B}) = s(\mathbf{B}, \mathbf{H}) + \beta s(\mathbf{B}, \mathbf{Q}), \tag{7}$$

where β represents the balancing factor. We opt to store the n_b memory blocks with the highest scores in the current key-value cache. In terms of intuition, $s(\mathbf{B}, \mathbf{Q})$ and $s(\mathbf{B}, \mathbf{H})$ respectively express the search for the memory blocks related to the query tokens $\mathbf{Q}$ and the current tokens $\mathbf{H}$. Ablation experiments in Sec. 4.4 validate that $\beta \geq 1$, indicating that the selection of queries is more crucial to the memory block. This aligns with our initial motivation. More methodology details are in Appendix A.

4 Experiments

In this section, we conduct experiments utilizing Mistral-7B-inst-v0.2 (Jiang et al., 2023) and LLaMA3-8B-inst (AI@Meta, 2024) as our base models. We compare our methods with LLaMA3-8B-inst-1048K (LLaMA-1048K) (Gradient, 2024) and other competing sliding window approaches,

Method Context Window	Infinite	Stream 512	ILM	QLLM	Infinite	Stream 1K	ILM	QLLM	Infinite	Stream 2K	ILM	QLLM
En.MC	26.64	27.95	26.63	**29.69**	31.00	30.13	33.19	**33.19**	30.13	30.57	33.62	**34.50**
Retrieve.PassKey	3.40	3.40	100.0	**100.0**	3.40	3.40	100.0	**100.0**	3.40	3.40	100.0	**100.0**
Retrieve.Number	3.39	3.39	99.73	**100.0**	3.39	3.39	99.83	**99.83**	3.39	3.39	**44.58**	40.00
Code.Debug	31.22	**32.74**	31.02	31.22	35.79	37.56	38.58	**38.58**	**35.79**	32.74	35.03	34.52
Math.Find	17.71	17.14	24.77	**25.93**	16.57	17.43	27.71	**28.86**	15.43	16.29	28.29	**28.29**
Retrieve.KV	0.20	0.20	13.18	**32.40**	0.40	0.40	32.60	**47.80**	1.00	1.00	62.94	**73.00**
Average	13.76	14.14	49.22	**53.21**	15.09	15.38	55.32	**58.04**	14.86	14.56	50.74	**51.72**

(a) ∞-Bench (214K tokens)

Method Context Window	Infinite	Stream 512	ILM	QLLM	Infinite	Stream 1K	ILM	QLLM	Infinite	Stream 2K	ILM	QLLM
NarrativeQA	8.80	9.77	11.80	**12.04**	9.44	10.19	15.61	**15.95**	12.44	13.37	18.75	**20.14**
Qasper	9.19	9.45	**16.13**	15.45	10.93	10.73	19.15	**19.24**	14.58	15.04	**20.78**	19.97
MultiFieldQA	25.38	26.03	38.43	**41.35**	27.82	27.76	42.65	**43.71**	32.29	32.02	43.74	**44.83**
HotpotQA	19.68	20.46	**28.19**	27.32	22.16	21.91	32.47	**34.47**	23.21	23.70	34.66	**36.53**
2WikiMQA	12.27	12.63	13.70	**15.22**	13.85	13.32	16.14	**16.57**	17.13	17.51	17.99	**19.97**
Musique	6.45	6.55	11.38	**12.99**	7.91	7.64	14.74	**15.27**	9.81	11.30	12.16	**17.05**
GovReport	22.50	22.40	**29.64**	28.46	24.79	24.90	**30.18**	29.82	27.07	27.12	**30.26**	29.75
QMSum	18.74	18.93	21.55	**21.69**	19.23	19.19	22.03	**22.27**	19.67	19.52	21.55	**22.36**
MultiNews	23.23	23.28	**25.19**	24.95	25.51	25.41	26.15	**26.39**	25.95	26.10	26.71	**26.84**
TREC	38.00	39.50	45.50	**47.50**	30.50	29.00	48.00	**49.50**	31.00	28.25	47.50	**48.25**
TriviaQA	79.68	80.54	82.02	**82.20**	**85.06**	84.27	83.20	84.56	**88.06**	87.08	82.81	84.49
SAMSum	35.30	34.58	36.65	**37.18**	36.05	35.09	**38.20**	38.12	36.30	36.21	37.91	**38.25**
PassageRetrieval	4.40	5.54	13.29	**25.04**	7.92	7.92	25.67	**31.04**	18.21	18.46	40.29	**49.67**
LCC	50.06	**51.59**	50.14	48.61	50.94	**53.27**	50.83	51.10	52.20	**54.95**	54.59	54.52
RepoBench-P	47.38	**48.04**	42.92	41.32	48.85	**51.31**	41.75	43.21	47.36	**47.60**	45.08	45.90
Average	25.18	25.59	29.30	**30.22**	26.40	26.44	31.80	**32.70**	28.56	28.77	33.55	**35.06**

(b) Long-Bench (31K tokens)

Table 1: The results comparison based on Mistral-7B-inst-v0.2 (Jiang et al., 2023). Our results are highlighted in teal and best results are indicated in bold.

containing LM-Infinite (Infinite) (Lin et al., 2024), StreamingLLM (Stream) (Xiao et al., 2024b) and InfLLM (ILM) (Xiao et al., 2024a). We test the methods on three cache lengths: 512, 1024 (1K), and 2048 (2K). More configuration details are in Appendix C. Note that we add the queries before the context to ensure the baselines also have query-aware capabilities, as detailed in Appendix E.

4.1 Long-Bench and ∞-Bench

In this section, we utilize representative tasks from two widely-recognized long document benchmarks, ∞-Bench (Zhang et al., 2024) and Long-Bench (Bai et al., 2023). The 95% quantile for sequence lengths in ∞-Bench and Long-Bench reaches 214K and 31K tokens. The outcomes based on Mistral-7B-inst-v0.2 and LLaMA3-8B-inst are detailed in Tab. 1 and Tab. 2 respectively. The following observations can be made from the results: (1) Our approach shows considerable enhancement in performance when compared to base models (LLaMA3-8B-inst-1048K) and that utilizing the sliding window mechanism (StreamingLLM and LM-Infinite) across benchmarks and context window lengths. This suggests that the context memory in QLLM can effectively provide LLMs with appropriate contextual data, facilitating efficient comprehension and reasoning on long sequences. (2) Our method also exhibits a significant performance uplift when compared to models with other lookup mechanisms (InfLLM). This implies that previous methods still struggle to extract valid information from noisy contexts. Our proposed Query-aware Context Lookup, however, can purposefully use the query to find relevant information in the long context. (3) Our technique is particularly beneficial in scenarios with longer input contexts and relatively smaller available context windows, as observed in comparisons across different benchmarks and context budgets. For instance, with a context window of 512 on the ∞-bench, QLLM improved by 7.17% compared to the current SOTA on LLaMA3, and by 3.26% on Mistral. This illustrates our model's superiority in infinite stream scenarios. (4) Our

Method	LLaMA	Infinite	Stream	ILM	QLLM	Infinite	Stream	ILM	QLLM	Infinite	Stream	ILM	QLLM
Context Window	-1048K	512				1K				2K			
En.MC	31.0	37.12	34.93	37.77	**40.17**	37.12	34.93	37.99	**40.17**	36.24	**37.12**	33.19	34.50
Retrieve.PassKey	6.78	3.40	3.40	100.0	**100.0**	3.40	3.40	100.0	**100.0**	3.40	3.40	100.0	**100.0**
Retrieve.Number	6.78	3.39	3.39	96.61	**98.98**	3.39	3.39	40.68	**41.19**	3.39	3.39	**28.64**	27.63
Code.Debug	22.59	22.59	22.59	22.59	**22.59**	22.59	22.59	23.10	**23.86**	**24.11**	22.84	22.59	23.10
Math.Find	34.29	20.86	19.71	29.23	**30.70**	20.86	19.71	**32.29**	31.14	18.00	16.86	26.86	**27.37**
Retrieve.KV	6.2	0.80	0.80	24.40	**61.20**	0.80	0.80	57.20	**70.80**	1.80	1.80	80.80	**84.00**
Average	17.94	14.69	14.14	51.77	**58.94**	14.69	14.14	48.54	**51.19**	14.49	14.23	48.68	**49.43**

(a) ∞-Bench (214K tokens)

Method	LLaMA	Infinite	Stream	ILM	QLLM	Infinite	Stream	ILM	QLLM	Infinite	Stream	ILM	QLLM
Context Window	-1048K	512				1K				2K			
NarrativeQA	23.78	14.50	14.56	19.28	**19.29**	14.50	14.56	19.90	**20.50**	16.47	15.12	19.41	**25.60**
Qasper	21.22	21.06	20.72	26.08	**26.58**	21.06	20.72	**32.35**	31.47	32.01	31.72	**41.27**	39.12
MultiFieldQA	39.89	25.66	25.79	36.01	**40.12**	25.66	25.79	41.46	**46.44**	31.63	30.99	45.89	**48.30**
HotpotQA	17.16	31.95	32.84	41.42	**42.34**	31.95	32.84	43.75	**49.15**	34.73	35.26	44.97	**49.91**
2WikiMQA	18.11	24.72	24.28	28.44	**29.63**	24.72	24.28	30.83	**31.53**	29.22	30.59	36.27	**39.63**
Musique	10.39	12.72	13.62	17.48	**18.75**	12.72	13.62	21.26	**23.95**	13.50	13.64	19.73	**25.03**
GovReport	33.76	26.25	25.93	**29.26**	26.83	26.25	25.93	**30.44**	28.73	27.84	27.83	**30.68**	29.80
QMSum	23.38	19.38	**19.42**	19.10	19.02	19.38	19.42	20.30	**20.62**	19.91	20.14	21.36	**22.23**
MultiNews	27.68	26.42	26.57	**26.63**	25.22	26.42	26.57	**27.46**	26.85	27.36	27.37	**27.87**	27.85
TriviaQA	87.76	82.46	**82.47**	80.81	77.65	82.46	82.47	**85.11**	85.04	**88.07**	87.35	88.03	87.70
SAMSum	41.89	38.28	37.91	38.73	**38.83**	38.28	37.91	39.59	**40.40**	**36.93**	35.97	34.86	34.97
PassageRetrieval	51.83	13.75	13.25	19.50	**35.00**	13.75	13.25	58.75	**69.00**	23.50	23.50	85.25	**88.00**
LCC	43.79	**54.17**	53.88	53.24	52.39	56.32	53.88	60.23	**60.43**	**60.42**	58.15	58.17	58.37
RepoBench-P	46.11	**60.71**	60.59	58.33	57.04	**62.81**	60.59	60.86	60.51	**64.95**	62.97	62.01	61.04
Average	34.77	32.29	32.27	35.31	**36.34**	32.59	32.27	40.88	**42.47**	36.18	35.76	43.98	**45.54**

(b) Long-Bench (31K tokens)

Table 2: The results comparison based on LLaMA3-8B-inst (AI@Meta, 2024). Our results are highlighted in teal and best results are indicated in bold.

model's relative improvement is more noticeable on LLaMA3 when compared to other models. This is because superior models can more effectively utilize query information to precisely locate relevant information in the long context.

4.2 Needle in a Haystack

Needle-in-a-Haystack (Kamradt, 2023) is a widely used benchmark to evaluate if models can effectively utilize extended context lengths. This test requires the model to accurately reproduce the details from a specific sentence (*needle*) that is randomly positioned within a document that could be as long as 128K (*haystack*). The results for methods based on Mistral-7B-inst-v0.2 and LLaMA3-8B-inst are shown in Fig. 5. The context window size is 512. Our method accurately locates the *needle* within the *haystack* across 1K to 128K tokens. Specifically, QLLM improved upon InfLLM by 7.0% on Mistral and achieved 100% on LLaMA3.

BABILong (Kuratov et al., 2024) is a challenging benchmark for evaluating the performance of models in processing arbitrarily long documents with distributed facts. We conducted experiments on the BABILong based on LLaMA3-8B-inst. At a window length of 1024, our method improved by 6.1% compared to InfLLM (Fig. 7).

4.3 Time and Memory Consumption

In Fig. 6(a) and (b), we compare the time and memory consumption of different input tokens across methods. QLLM and InfLLM are comparable in terms of efficiency and memory usage. The time consumed by InfLLM and QLLM increases almost linearly with the number of input tokens, requiring only 25.6 seconds and 22.3GB of memory to process 100k tokens. In contrast, LLaMA3-8B-inst-1048K shows a rapid increase in time and memory consumption with the number of input tokens and cannot handle 16k tokens on a single A800 GPU (maximum memory of 80GB). The context length for InfLLM and QLLM is 2048.

4.4 Ablation Experiments

To further substantiate the efficacy of Query-aware Context Lookup, we carry out ablation studies in

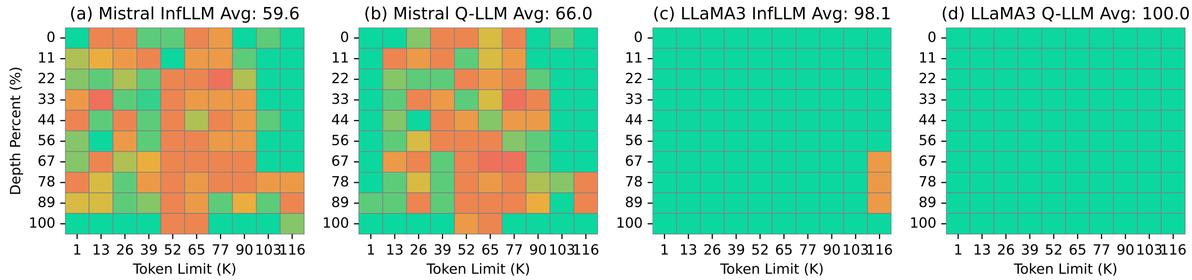


Figure 5: The comparison of performance in the Needle-in-a-Haystack task. The horizontal axis represents the document's length (the *haystack*), whereas the vertical axis specifies the location of a brief sentence (the *needle*) within the document, ranging from 1K to 128K tokens. A red cell indicates the language model's inability to recall the needle's information, while a green cell denotes successful recall by the model.

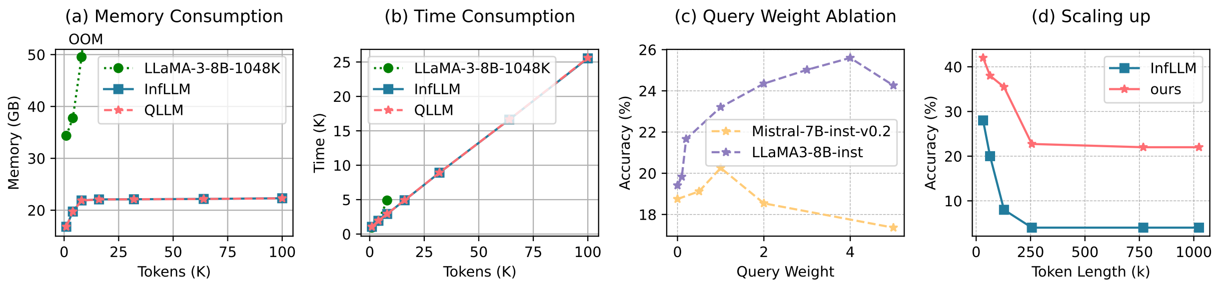


Figure 6: (a) Memory and (b) Time consumption of different methods as tokens increases. (c) Ablation of query weight β. (d) The results of methods on sequences with extremely lengthy sequence lengths.

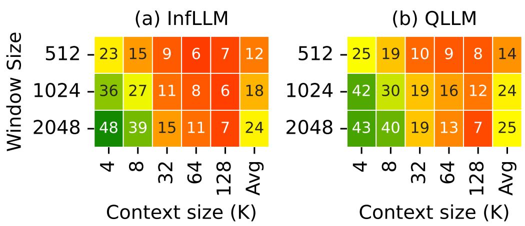


Figure 7: Average accuracy over QA1-QA5 tasks from BABILong. The horizontal and vertical axes represent the context and window length.

this section, with results displayed in Fig. 6(c). The performance of Mistral-7B-inst-v0.2 and LLaMA3-8B-inst exhibits a trend of initial increase followed by a decrease as the query weight β escalates. We select the β at the peak point as the experimental setup, choosing $\beta = 1$ for Mistral and $\beta = 4$ for LLaMA3. Further exploration of the primary elements within the context memory is in Appendix B.

4.5 Scaling up

In this sub-section, we're evaluating QLLM's capacity to handle extremely lengthy sequences by extending the sequence length to 1024K. The base model used is Mistral-7B-inst-v0.2 and the task is *Retrieve.KV* from ∞-Bench. The outcomes are displayed in Fig. 6(d). According to the results, even when the context length is scaled to 1024 thousand tokens, QLLM consistently performs at a level significantly above the current SOTA. This confirms QLLM's ability to accurately recognize long-distance dependencies for effective long-sequence reasoning.

5 Conclusion

In this study, we focused on the significant challenges faced by LLMs in processing and reasoning over extensive contexts. We introduced QLLM, an approach inspired by human cognitive processes, which focuses on relevant memory data and effectively bypasses context input clutter. Our method does not require additional training and can be seamlessly integrated with any LLM. Through comprehensive evaluations using the LLaMA and Mistral models on the Longbench, ∞-Bench, Needle-in-a-Haystack and BABILong benchmarks, QLLM demonstrated a marked improvement over the current SOTA. Moreover, our QuickLLaMA can read 100K tokens within 30 seconds. The empirical results validate QLLM's ability to capture long-range dependencies and manage vast contexts efficiently, paving the way for enhanced performance in LLM-driven tasks that require long-sequence reasoning.

Limitations

While QLLM demonstrates promising improvements over the current SOTA methods in various benchmarks, there are still some limitations. For instance, the system's performance relies on the limited window size, which could lead to potential information loss when dealing with highly complex contexts. Future research should address these limitations and explore the potential of QLLM in a broader range of tasks and contexts.

Broader Impact

The advancements made by QLLM in understanding and reasoning over broad contexts, a long-term research focus of Large Language Models (LLMs), could have profound implications across various fields. Given its ability to manage lengthy sequences,QLLM's potential to operate consistently over the content of conversations spanning recent days could make ChatBot assistants more effective and user-friendly. Tasks such as summarizing and answering questions based on books, reports, and documents, as well as generating code at the repository level, could also be improved with the ability to handle long context sequences. However, it is crucial to recognize that the benefits of QLLM also come with potential risks. The system's ability to process and understand extensive contexts could be misused for nefarious purposes, such as creating deepfakes or other forms of misinformation.

References

AI@Meta. 2024. Llama 3 model card.

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. Etc: Encoding long and structured inputs in transformers. *Preprint*, arXiv:2004.08483.

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

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

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

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. 2023. CLEX: continuous length extrapolation for large language models. *CoRR*, abs/2310.16450.

Guanzheng Chen, Xin Li, Zaiqiao Meng, Shangsong Liang, and Lidong Bing. 2024. Clex: Continuous length extrapolation for large language models. *Preprint*, arXiv:2310.16450.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *Preprint*, arXiv:1901.02860.

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

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Preprint*, arXiv:2205.14135.

Zican Dong, Tianyi Tang, Lunyi Li, and Wayne Xin Zhao. 2023. A survey on long text modeling with transformers. *Preprint*, arXiv:2302.14502.

Gradient. 2024. Llama-3-8b-instruct-gradient-1048k.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *Preprint*, arXiv:1410.5401.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces. *Preprint*, arXiv:2312.00752.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. *Preprint*, arXiv:2111.00396.

Ke Hong, Guohao Dai, Jiaming Xu, Qiuli Mao, Xiuhong Li, Jun Liu, Kangdi Chen, Yuhan Dong, and Yu Wang. 2024. Flashdecoding++: Faster large language model inference on gpus. *Preprint*, arXiv:2311.01282.

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. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. 2024. LLM maybe longlm: Self-extend LLM context window without tuning. *CoRR*, abs/2401.01325.

Greg Kamradt. 2023. Needle in a haystack - pressure testing llms. https://github.com/gkamradt/LLMTest_NeedleInAHaystack.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention. *Preprint*, arXiv:2006.16236.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *Preprint*, arXiv:2001.04451.

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. *Preprint*, arXiv:2309.06180.

Jingyao Li, Pengguang Chen, and Jiaya Jia. 2024a. Motcoder: Elevating large language models with modular of thought for challenging programming tasks. *Preprint*, arXiv:2312.15960.

Jingyao Li, Pengguang Chen, Shengju Qian, and Jiaya Jia. 2023a. Tagclip: Improving discrimination ability of open-vocabulary semantic segmentation. *Preprint*, arXiv:2304.07547.

Jingyao Li, Pengguang Chen, Sitong Wu, Chuanyang Zheng, Hong Xu, and Jiaya Jia. 2024b. Robocoder: Robotic learning from basic skills to general tasks with large language models. *Preprint*, arXiv:2406.03757.

Jingyao Li, Pengguang Chen, Shaozuo Yu, Zexin He, Shu Liu, and Jiaya Jia. 2023b. Rethinking out-of-distribution (ood) detection: Masked image modeling is all you need. *arXiv preprint arXiv:2302.02615*.

Jingyao Li, Pengguang Chen, Shaozuo Yu, Shu Liu, and Jiaya Jia. 2023c. Bal: Balancing diversity and novelty for active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–12.

Jingyao Li, Pengguang Chen, Shaozuo Yu, Shu Liu, and Jiaya Jia. 2024c. Moodv2: Masked image modeling for out-of-distribution detection. *Preprint*, arXiv:2401.02611.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. 2024d. Snapkv: Llm knows what you are looking for before generation. *Preprint*, arXiv:2404.14469.

Bin Lin, Tao Peng, Chen Zhang, Minmin Sun, Lanbo Li, Hanyu Zhao, Wencong Xiao, Qi Xu, Xiafei Qiu, Shen Li, Zhigang Ji, Yong Li, and Wei Lin. 2024. Infinite-llm: Efficient llm service for long context with distattention and distributed kvcache. *Preprint*, arXiv:2401.02669.

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. *CoRR*, abs/2307.03172.

Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1400–1409. The Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. *Preprint*, arXiv:2108.12409.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Noam Shazeer. 2019. Fast transformer decoding: One write-head is all you need. *CoRR*, abs/1911.02150.

Jianlin Su. 2023. Rectified rotary position embeddings. `https://github.com/bojone/rerope`.

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. Roformer: Enhanced transformer with rotary position embedding. *Preprint*, arXiv:2104.09864.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2440–2448.

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. 2024a. Focused transformer: Contrastive training for context scaling. *Advances in Neural Information Processing Systems*, 36.

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. 2024b. Focused transformer: Contrastive training for context scaling. *Advances in Neural Information Processing Systems*, 36.

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

Jason Weston, Sumit Chopra, and Antoine Bordes. 2015. Memory networks. *Preprint*, arXiv:1410.3916.

Yuhuai Wu, Markus N. Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing transformers. *Preprint*, arXiv:2203.08913.

Chaojun Xiao, Pengle Zhang, Xu Han, Guangxuan Xiao, Yankai Lin, Zhengyan Zhang, Zhiyuan Liu, Song Han, and Maosong Sun. 2024a. Infllm: Unveiling the intrinsic capacity of llms for understanding extremely long sequences with training-free memory. *Preprint*, arXiv:2402.04617.

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

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2021. Big bird: Transformers for longer sequences. *Preprint*, arXiv:2007.14062.

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

Guangxiang Zhao, Junyang Lin, Zhiyuan Zhang, Xuancheng Ren, Qi Su, and Xu Sun. 2019. Explicit sparse transformer: Concentrated attention through explicit selection. *CoRR*, abs/1912.11637.

Liang Zhao, Xiaocheng Feng, Xiachong Feng, Bing Qin, and Ting Liu. 2023. Length extrapolation of transformers: A survey from the perspective of position encoding. *CoRR*, abs/2312.17044.

Chuanyang Zheng, Yihang Gao, Han Shi, Minbin Huang, Jingyao Li, Jing Xiong, Xiaozhe Ren, Michael Ng, Xin Jiang, Zhenguo Li, et al. 2024. Cape: Context-adaptive positional encoding for length extrapolation. *arXiv preprint arXiv:2405.14722*.

A Methodology Details

In this section, we introduce our methodology details.

A.1 Chunks

Given the constraints of GPU memory, we do not encode the input sequence at once (Xiao et al., 2024a); instead, we process it in chunks and generate output on a token-by-token basis. In each computational step, the inputs are composed of past key-value vectors $\mathbf{P} = \{(\mathbf{k}_i, \mathbf{v}_i)\}_{i=1}^{l_P}$ and current token hidden vectors $\mathbf{H} = \{\mathbf{h}_i\}_{i=1}^{l_H}$. When encoding, l_H is equivalent to the chunk size, while during decoding, l_H is equal to one.

A.2 Positional Encoding

Traditional LLM training typically utilizes a limited set of positional encodings, which can face difficulties with out-of-domain distribution when extended to process longer sequences (Xiao et al., 2024a). Furthermore, in QLLM, the current key-value cache consists of several discontinuous text blocks. Assigning continuous positional encodings to these blocks could create mismatches and confuse the model. Consequently, drawing upon previous studies (Raffel et al., 2023; Su, 2023; Xiao et al., 2024a), we assign identical positional encodings to all tokens exceeding the local window size. More precisely, we set the distance between tokens in context memory blocks and the current tokens as l_L.

A.3 Cache Management

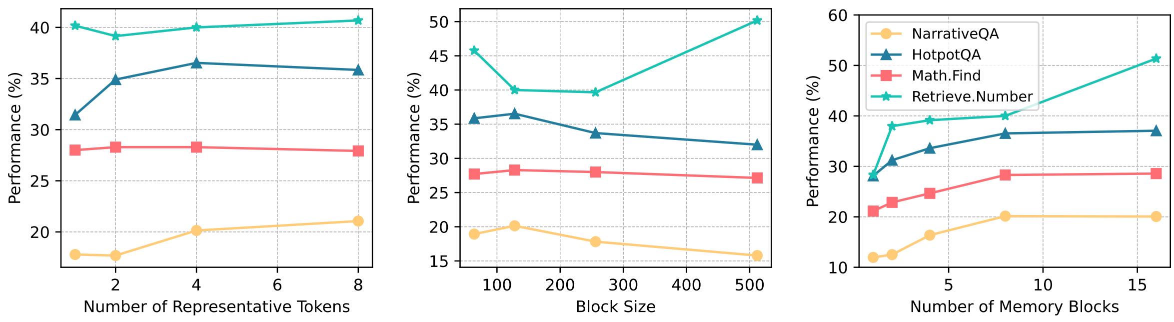
In order to process exceedingly lengthy sequence streams and encapsulate the semantic relevance with LLMs (Xiao et al., 2024a), it's necessary to maintain all memory blocks and reference them at every computational stage. Given that most blocks are seldom used, we adopt an offloading strategy, which stores most memory blocks in CPU memory. Only the tokens and memory blocks essential for current operations are kept in GPU memory. Furthermore, due to the semantic continuity in long sequences where neighboring tokens often necessitate similar memory blocks, we assign a cache area in GPU memory, governed by a least recently used policy. This method enables efficient encoding of exceptionally long sequences using finite GPU memory. Moreover, for extremely long sequences, the representative tokens for each block can be offloaded to CPU memory, forming an effective k-nearest-neighbor index, thereby further diminishing computational complexity.

B Further Exploration

QLLM leverages context memory to retrieve pertinent data. We delve deeper into the influence of primary elements within the context memory. Results are presented in Fig. 8. Conduct experiments on Mistral-7B-inst-v0.2 using the default parameters with a context window length of 1024.

B.1 Number of Representative Tokens

QLLM divides key-value vectors into memory blocks and picks a few representative tokens from each block to act as the segment's representation. The capacity of these tokens to symbolize the entire segment semantically directly impacts the model's efficacy. We run tests with the different quantity of representative tokens. The outcomes are displayed in Fig. 8a. We note an upward trend in model performance as the number of tokens increases, suggesting that a larger token count can better capture the semantic essence of memory segments. However, when the token count hits 8, a slight performance dip is seen in HotpotQA. This drop can be traced back to the inclusion of semantically unrelated tokens in the segment representations. Future work could enhance model performance by



(a) Representative Tokens (b) Block Size (c) Number of Blocks

Figure 8: Further exploration, investigating the influence of the context memory with varying numbers of representative tokens, selected memory blocks, and memory block sizes, respectively.

developing more efficient and potent segment representations.

B.2 Memory Block Size

Each memory block should ideally represent a consistent semantic block. Oversized blocks can obstruct precise lookup, while undersized ones can escalate the computational cost of memory lookup. We test QLLM with different block sizes while maintaining a total context length of 1024. The outcomes are displayed in Fig. 8b. It's evident that the best block size changes according to the task due to the differing characteristics of input sequences. For instance, in NarrativeQA, a semantically complete paragraph forms a semantic block, while in Retrieve.Number, a single number does. Using heuristic rules to segment context can result in less-than-optimal performance. Thus, investigating how to dynamically segment context is a vital future research direction.

B.3 Number of Memory blocks

The chosen blocks are employed to supply relevant context to LLMs. We run tests with different number of blocks. From Fig. 8c, we see that as the number of chosen blocks rises, there's a significant improvement in model performance. This is because more blocks mean a higher recall rate of relevant content. However, a larger block count also increases the time needed for memory scheduling and the computational time for attention. Hence, advancing lookup accuracy continues to be a key area for enhancing the efficiency of LLMs.

Context Window	Local Tokens	Block Size	Block Num
512	256	64	4
1024	512	64	8
2048	1024	128	8

Table 3: The parameters for different context window length, including number of local tokens, memory block size and number of memory blocks.

C Configuration Details

C.1 Datasets

We utilize representative tasks from following widely-recognized long document benchmarks.

(1) **∞-Bench (Zhang et al., 2024) and LongBench (Bai et al., 2023)**. Given that our base models are primarily pre-trained on English corpora, we employ English datasets for the evaluation. These benchmarks encompass a variety of tasks such as question answering, summarization, few-shot learning, context retrieval, mathematical computing, and code completion. The average document length in ∞-Bench is 145.1K tokens, and in LongBench, it is 12.8K tokens. The 95% quantile for sequence lengths in these benchmarks reaches 214K and 31K tokens respectively, which significantly exceeds the maximum length of the base models.

(2) **Needle-in-a-Haystack** (Kamradt, 2023), a widely used benchmark to evaluate if models can effectively use extended context lengths. This test requires the model to accurately reproduce the details from a specific sentence (referred to as the *needle*) that is randomly positioned within a document that could be as long as 128K (referred to as the *haystack*). We adopted the following setting:

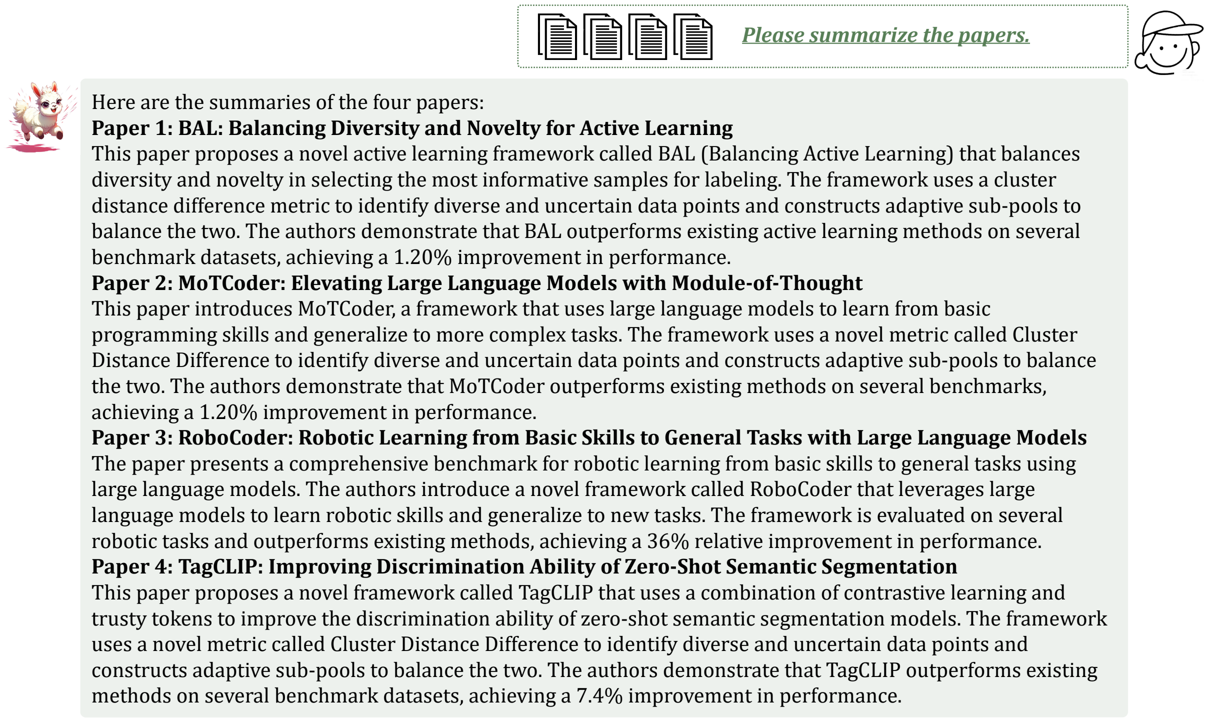


Figure 9: Examples of our QuickLLaMA-8B summarizing multiple papers (Li et al., 2023c, 2024a, 2023a, 2024b).

the needle is *The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.*, and the haystack is *PaulGrahamEssays*. The retrieval question is *What is the best thing to do in San Francisco?*.

(3) **Scaling up**. To evaluate QLLM's capacity to handle extremely lengthy sequences by extending the sequence length to 1024K. We use the *Retrieve.KV* task from the ∞-Bench to test its ability to discern context in extensive sequences. This task requires LLMs to identify a value from a key and a dictionary, essentially locating pertinent information within long sequences. Inputs with $\{32, 64, 128, 256, 768, 1024\}$ thousand tokens are automatically generated. For each length, 50 instances are created for assessment.

(4) **BABILong** is a challenging benchmark for evaluating the performance of models in processing arbitrarily long documents with distributed facts. BABI tasks are generated by simulating a set of characters and objects engaged in various movements and interactions with each other in multiple locations. Each interaction is represented by a fact, and the task is to answer a question using the facts from the current simulation. The BABI tasks vary based on the number of facts, question complexity and the aspects of reasoning.

C.2 Baseline Methods

Our goal is to enable LLMs trained with limited sequence lengths to comprehend extremely long sequences without additional training. For this purpose, we use Mistral-7B-Instruct-v0.2 (Jiang et al., 2023) and LLaMA3-8B-inst (AI@Meta, 2024) as our base models. Mistral-7B-Instruct-v0.2 is initially pre-trained with a maximum sequence length of 8K tokens and subsequently fine-tuned with a maximum sequence length of 32K tokens. LLaMA3-8B-inst is fine-tuned from LLaMA3-8B (AI@Meta, 2024) with an 8K token maximum length. We compare our methods with competing sliding window approaches, containing LM-Infinite (Infinite) (Lin et al., 2024), StreamingLLM (Stream) (Xiao et al., 2024b) and InfLLM (infllm) (Xiao et al., 2024a). The same prompts are utilized for all methods and detailed in Appendix E.

C.3 Implementation Details

Following the settings of previous works (Xiao et al., 2024a), the number of representative tokens, r_k, is set to 4. The number of initial tokens is set to 128 for LM-Infinite, StreamingLLM, InfLLM, and QLLM, to cover system prompts and task descriptions. We test the methods on three cache lengths: 512, 1024 (1K), and 2048 (2K). The local tokens, block size, and number of blocks for InfLLM and

QLLM are detailed in Tab. 3.

D More Examples

In this section, we provide examples of our QuickLLaMA-8B.

1. Figure 9 summarizes multiple papers.
2. Figure 10 summarizes paper series and discovering their connection and comparison.
3. Figure 11 reviews our paper according to ACL review guideline.
4. Figure 12 reads long novel *Journey to the West* comparing with current SOTAs.

E Prompts

We utilize the same prompts for all experiments in the main paper.

1. The prompts for Long-Bench are in Tabs. 4 and 5.
2. The prompts for ∞-Bench are in Tab. 6.
3. The prompts for Needle-in-a-Haystack Bench are in Tab. 7.

Please summarize the two papers.

Here are the summaries of the two papers:
Paper 1:
The paper proposes a new approach to out-of-distribution (OOD) detection called Masked Image Modeling for OOD detection (MOOD). The authors argue that previous methods that use recognition-based methods to learn in-distribution (ID) features tend to learn shortcuts instead of comprehensive representations. In contrast, they show that using reconstruction-based methods can significantly improve OOD detection performance. The authors use masked image modeling as a pretext task to learn ID features and achieve state-of-the-art results on one-class OOD detection, multi-class OOD detection, near-distribution OOD detection, and few-shot outlier exposure OOD detection.
Paper 2:
The paper explores the importance of effective out-of-distribution (OOD) detection in visual recognition systems. The authors argue that previous methods often rely on recognition-based methods to learn in-distribution (ID) features, which can lead to shortcut learning. Instead, they propose using reconstruction-based pretext tasks to learn ID features. The authors conduct experiments with different pretext tasks and OOD score functions and find that feature representations pre-trained through reconstruction can significantly enhance performance and reduce the performance gap among different score functions. They introduce the MOODv2 OOD detection framework, which uses masked image modeling as a pretext task and achieves state-of-the-art results on ImageNet and CIFAR-10.

What is the connection between the papers?

The connection between the two papers is that they both deal with the topic of Out-of-Distribution (OOD) detection, which is a crucial task in machine learning. The first paper introduces a new approach called MOOD (Masked Out-of-Distribution Detection) that uses a reconstruction-based pretext task to learn a robust in-distribution (ID) representation, which is distinguishable from OOD samples. The paper reports state-of-the-art results on various OOD detection tasks.

The second paper, MOODv2, is an extension of the first paper, and it further explores the idea of using reconstruction-based pretext tasks for OOD detection. The paper conducts a comprehensive analysis of different pretraining tasks and OOD score functions, and finds that feature representations pre-trained through reconstruction can significantly enhance performance and reduce the performance gap among different score functions. The paper introduces a new OOD detection framework, MOODv2, which uses the masked image modeling pretext task and achieves state-of-the-art results on various OOD detection tasks.

In summary, the two papers are connected in that they both focus on OOD detection and explore the use of reconstruction-based pretext tasks to learn robust ID representations. The second paper builds upon the ideas presented in the first paper and provides a more comprehensive analysis and a new OOD detection framework.

Please compare the papers and summarize the improvement of the second paper from the first paper.

The first paper introduces a novel approach to out-of-distribution (OOD) detection using masked image modeling as a pretext task, which is called MOOD. The paper shows that MOOD outperforms previous state-of-the-art (SOTA) methods on one-class, multi-class, near-distribution, and few-shot outlier exposure OOD detection tasks.

The second paper, MOODv2, builds upon the first paper and makes several improvements. The main contributions of MOODv2 are:
1. **Improved AUROC**: MOODv2 achieves a remarkable 14.30% increase in AUROC on ImageNet, reaching 95.68%, and a 3.56% increase on CIFAR-10, reaching 98.20%.
2. **Reduced FPR95**: MOODv2 reduces the FPR95 by 44.93% on ImageNet and 15.14% on CIFAR-10.
3. **Improved performance on multi-class OOD detection**: MOODv2 achieves an impressive AUROC of 99.98% on CIFAR-10 and 95.68% on ImageNet, outperforming previous SOTA methods.
4. **Robustness to different score functions**: MOODv2 shows that even simple score functions can perform as well as complex ones when using reconstruction-based pretext tasks, indicating the robustness of the approach.

The improvements in MOODv2 can be attributed to the following:
1. **Comprehensive analysis**: MOODv2 conducts a comprehensive analysis of different pretraining tasks and OOD score functions, which helps to identify the most effective combination.
2. **Improved pretext task**: MOODv2 uses the masked image modeling pretext task, which is more effective than the original masked image modeling task used in the first paper.
3. **Better feature representation**: MOODv2's feature representation is more robust and effective, which is reflected in the improved AUROC and reduced FPR95.

Overall, MOODv2 builds upon the foundation laid by the first paper and makes significant improvements in terms of performance, robustness, and effectiveness.

Figure 10: Examples of our QuickLLaMA-8B summarizing paper series and discovering their connection and comparison (Li et al., 2023b, 2024c).

Please review the paper based on the review guideline. Below is the review guideline for the Association for Computational Linguistics (ACL). {review guideline}

Paper Summary. The paper presents QuickLLaMA (Q-LLM), a query-aware inference acceleration method for large language models (LLMs). The authors aim to address the challenge of processing and reasoning over extensive contexts by focusing on relevant memory data and bypassing context input clutter. Q-LLM is designed to be seamlessly integrated with any LLM and does not require additional training. The paper evaluates Q-LLM using the LLaMA and Mistral models on the Longbench and ∞-Bench benchmarks, demonstrating a marked improvement over the current state-of-the-art, particularly in handling sequences of significant length.
Summary of Strengths. 1. The paper addresses a saignificant challenge in the field of LLMs, which is the ability to process and reason over extensive contexts. 2. The proposed method, Q-LLM, is novel and innovative, and its ability to focus on relevant memory data and bypass context input clutter is a significant contribution. 3. The paper provides comprehensive evaluations using multiple benchmarks, including the Longbench and ∞-Bench, which demonstrates the effectiveness of Q-LLM. 4. The authors provide a clear and concise explanation of the method and its advantages, making it easy to understand for readers.
Summary of Weaknesses. 1. The paper assumes that the input sequences are long and complex, which may not be the case for all applications. It would be beneficial to discuss the limitations of the method and potential future work to address these limitations. 2. The paper does not provide a detailed analysis of the computational complexity of Q-LLM, which may be important for large-scale applications. 3. The paper could benefit from more discussion on the potential applications of Q-LLM, beyond the specific benchmarks used in the evaluation.

Comments/Suggestions/Typos. 1. The paper could benefit from a more detailed introduction to the problem of processing and reasoning over extensive contexts, including a review of the current state-of-the-art methods. 2. The authors could provide more information on the training data and hyperparameters used for the LLaMA and Mistral models. 3. The paper could benefit from more visualizations, such as plots or diagrams, to help illustrate the results and make them more accessible to readers.
Soundness. I would rate the soundness of this study as 4.5 out of 5. The paper provides a clear and concise explanation of the method and its advantages, and the evaluations using multiple benchmarks demonstrate the effectiveness of Q-LLM. However, the paper assumes that the input sequences are long and complex, which may not be the case for all applications, and the computational complexity of Q-LLM is not fully discussed.
Overall Assessment. I would rate the overall assessment of this paper as 4.5 out of 5. The paper presents a novel and innovative method for processing and reasoning over extensive contexts, and the evaluations demonstrate its effectiveness. However, the paper could benefit from more discussion on the limitations and potential applications of Q-LLM, as well as more visualizations to help illustrate the results.
Best Paper. I would not recommend this paper for an "outstanding paper " award, but it is a strong contender for publication at an ACL event.
Reviewer Confidence. I am confident in my evaluation, having carefully read the paper and evaluated its strengths and weaknesses.

Figure 11: Examples of our QuickLLaMA-8B reviewing our paper according to ACL review guideline.

Read the book and answer my question: ***Why Sun Wukong presents undefeated strength when he caused havoc in heaven while he waslatter defeated by many monsters during the journey to the west?***

According to the book, Sun Wukong, also known as the Monkey King, presents undefeated strength when he caused havoc in heaven because he was a powerful and arrogant immortal who had not yet cultivated his mind and had not yet understood the concept of the Buddhist teachings. He was a rebellious and mischievous being who enjoyed causing trouble and chaos. However, during the Journey to the West, Sun Wukong is defeated by many monsters and demons because he has undergone a transformation. He has been ***converted to Buddhism and has learned to cultivate his mind and understand the concept of the Buddhist teachings***. He has also learned to use his powers for good and to protect his friends and companions. In other words, ***Sun Wukong's undefeated strength in heaven was a result of his arrogance and lack of understanding***, while his defeats during the Journey to the West were a result of his growth and development as a character. He has learned to use his powers wisely and to understand the importance of compassion and self-discipline. It's also worth noting that Sun Wukong's defeats during the Journey to the West are not necessarily a reflection of his physical strength, but rather a reflection of his spiritual growth and development. ***He is still a powerful and formidable character, but he has learned to use his powers in a more responsible and compassionate way.***

InfLLM

According to the book, Sun Wukong, also known as the Monkey King, presents undefeated strength when he causes havoc in heaven because he is a powerful and arrogant immortal ***who has mastered the art of magic*** and has the ability to transform himself into various forms. He is confident in his abilities and believes that he is invincible. However, during the journey to the West, he is defeated by many monsters and demons because he is not as powerful as he thinks he is. He is also reckless and impulsive, which often gets him into trouble. Additionally, he is not as skilled in the art of magic as he thinks he is, and his powers are not as effective against the monsters and demons they encounter. In particular, his defeat at the hands of the Demon King of the Underworld, who is also his uncle, is a significant turning point in the story. The Demon King is able to defeat Sun Wukong because he has mastered the art of magic and has the power to control the elements. ***Sun Wukong,*** on the other hand, ***is unable to defeat the Demon King because he is not as powerful or skilled in magic***. Overall, the book suggests that Sun Wukong's undefeated strength in heaven is an illusion, and that he is not as powerful as he thinks he is. His defeats during the journey to the West serve as a humbling experience for him, and he learns to be more humble and respectful of others.

Stream LLM

According to the book, Sun Wukong, also known as the Monkey King, presents undefeated strength when he causes havoc in heaven because he is a powerful and arrogant demon who has not yet been defeated. He is able to wreak havoc in heaven because he is not yet bound by the rules of the Buddhist scriptures and is able to use his magical powers to defy the gods. However, during the journey to the West, Sun Wukong is defeated by many monsters and demons because he is no longer invincible. He is forced to confront the consequences of his own arrogance and is humbled by his experiences. ***He is defeated by the Red Boy, the White Bone Demon, and other powerful enemies, and is forced to flee and hide.*** The reason for this change in Sun Wukong's fortunes is that he has been bound by the Buddhist scriptures and is no longer able to use his magical powers to defy the gods. He is also forced to confront the consequences of his own arrogance and is humbled by his experiences.

LM-Infinite

According to the book, Sun Wukong, also known as the Monkey King, presents undefeated strength when he causes havoc in heaven because he is a powerful and arrogant demon who has not yet been defeated. He is able to wreak havoc in heaven because he is not yet bound by the rules of the Buddhist scriptures and is able to use his magical powers to defy the gods. However, during the journey to the West, Sun Wukong is defeated by many monsters and demons because he is no longer invincible. He is forced to confront the consequences of his own arrogance and is humbled by his experiences. ***He is defeated by the Red Boy, the White Bone Demon, and other powerful enemies, and is forced to flee and hide.*** The reason for this change in Sun Wukong's fortunes is that he has been bound by the Buddhist scriptures and is no longer able to use his magical powers to defy the gods. He is also forced to confront the consequences of his own arrogance and is humbled by his experiences.

Figure 12: Examples of our QuickLLaMA-8B reading long novel *Journey to the West* comparing with current SOTAs

Dataset	Prompt
NarrativeQA	You are given a story, which can be either a novel or a movie script, and a question. Answer the question asconcisely as you can, using a single phrase if possible. Do not provide any explanation. Question: {input} Story: {context} Now, answer the question based on the story asconcisely as you can, using a single phrase if possible. Do not provide any explanation. Question: {input} Answer:
Qasper	You are given a scientific article and a question. Answer the question as concisely as you can, using a single phrase or sentence if possible. If the question cannot be answered based on the information in the article, write "unanswerable". If the question is a yes/no question, answer "yes", "no", or "unanswerable". Do not provide any explanation. Question: {input} Article: {context} Answer the question based on the above article as concisely as you can, using a single phrase or sentence if possible. If the question cannot be answered based on the information in the article, write "unanswerable". If the question is a yes/no question, answer "yes", "no", or "unanswerable". Do not provide any explanation. Question: {input} Answer:
MultiFieldQA	Read the following text and answer briefly. Question: {input} {context} Now, answer the following question based on the above text, only give me the answer and do not output any other words. Question: {input} Answer:
HotpotQA	Answer the question based on the given passages. Only give me the answer and do not output any other words. The following are given passages. Question: {input} {context} Answer the question based on the given passages. Only give me the answer and do not output any other words. Question: {input} Answer:
2WikiMQA	Answer the question based on the given passages. Only give me the answer and do not output any other words. The following are given passages. Question: {input} {context} Answer the question based on the given passages. Only give me the answer and do not output any other words. Question: {input} Answer:
Musique	Answer the question based on the given passages. Only give me the answer and do not output any other words. The following are given passages. Question: {input} {context} Answer the question based on the given passages. Only give me the answer and do not output any other words. Question: {input} Answer:
GovReport	You are given a report by a government agency. Write a one-page summary of the report . Report: {context} Now, write a one-page summary of the report. Summary:

Table 4: The prompt for each dataset in Long-Bench (Bai et al., 2023). Yellow highlights indicate the query, while a gray background represents the long context.

Dataset	Prompt
QMSum	You are given a meeting transcript and a query containing a question or instruction. Answer the query in one or more sentences. Query: {input} Transcript: {context} Now, answer the query based on the above meeting transcript in one or more sentences. Query: {input} Answer:
MultiNews	You are given several news passages. Write a one-page summary of all news . News: {context} Now, write a one-page summary of all the news. Summary:
TREC	Please determine the type of the question below. {input} Here are some examples of questions. {context} Now please determine the type of the question below. {input}
TriviaQA	Answer the question based on the given passage. Only give me the answer and do not output any other words. {input} The following are some examples. {context} Now answer the question based on the given passage. Only give me the answer and do not output any other words. {input}
SAMSum	Summarize the dialogue into a few short sentences. {input} The following are some examples. {context} {input}
PassageRetrieval	Here are 30 paragraphs from Wikipedia, along with an abstract. Please determine which paragraph the abstract is from. Abstract: {input} Paragraphs: {context} Please enter the number of the paragraph that the abstract is from. The answer format must be like "Paragraph 1", "Paragraph 2", etc. The answer is:
LCC	Please complete the code given below . {context} Next line of code:
RepoBench-P	Please complete the code given below . {context} {input} Next line of code:

Table 5: The prompt for each dataset in Long-Bench (Bai et al., 2023) (continued). Yellow highlights indicate the query, while a gray background represents the long context.

Dataset	Prompt
En.MC	Read the book and answer the question.Only one of the following options is correct, tell me the answer using one single letter (A, B, C, or D). Don't say anything else. Question: {question} A. {OPTIONA} B. {OPTIONB} C. {OPTIONC} D. {OPTIOND} {context} Question: {question} Only one of the following options is correct, tell me the answer using one single letter (A, B, C, or D). Don't say anything else. A. {OPTIONA} B. {OPTIONB} C. {OPTIONC} D. {OPTIOND}
Retrieve.PassKey	There is an important info hidden inside a lot of irrelevant text. Find and memorize it: {input} {context} {input}
Retrieve.Number	There is an important info hidden inside a lot of irrelevant text. Find and memorize it: {input} {context} {input}
Code.Debug	There is ONLY ONE function in the large project that is deliberately made to include an obvious error. Please find the function that contains the most obvious errors. I will give you four options to narrow your scope. You can inspect the options and think. Eventually, tell me the answer using one single letter (A, B, C, or D). Which funtion has deliberate error? A. {OPTIONA} B. {OPTIONB} C. {OPTIONC} D. {OPTIOND} {context} Which funtion has deliberate error? A. {OPTIONA} B. {OPTIONB} C. {OPTIONC} D. {OPTIOND} Give me your answer for the function that has the deliberate and obvious error in A, B, C, or D. Your answer MUST be chosen from one of the four options without any explanation. If you cannot determine answers accurately, you also MUST provide the answer you think is most likely. Absolutely do not say you do not know or you need more information.
Math.Find	{prefix} {context} {input}
Retrieve.KV	Extract the value corresponding to the specified key {key} in the JSON object below. {context} {input}

Table 6: The prompt for each dataset in ∞-Bench (Zhang et al., 2024) (continued). Yellow highlights indicate the query, while a gray background represents the long context.

Dataset	Prompt
Needle in a Haystack	Based on the content of the book, Question: What is the best thing to do in San Francisco? <book> <context> </book>.

Table 7: The prompt for each dataset in Needle-in-a-Haystack Benchmark (Kamradt, 2023). Yellow highlights indicate the query, while a gray background represents the long context.