Journal of Computer Technology and Software

ISSN:2998-2383

Vol. 3, No. 6, 2024

Learning What to Remember: Key Signal-Aligned Memory Compression for Long-Context LLMs

Junqi Zhang[1*]
[1]Columbia University, New York, USA
*Corresponding author: Junqi Zhang; junqizhang66@gmail.com

Abstract: This paper addresses the problems of information redundancy, sparsity of key clues, and difficulty in cross-round referencing in long-context scenarios, proposing a unified framework combining memory compression and key information fine-tuning. The method first decomposes the input into controllable-granularity fragments and encodes them. A lightweight scoring mechanism is used to assign importance weights to these fragments. Under a fixed memory budget, an explicit memory set is constructed through a selection strategy, and a global memory vector is formed through weighted aggregation to supplement the overall semantics and reduce the risk of omissions. To align memory selection with model behavior, a key supervision signal is further introduced to constrain importance estimation, and key-related objectives are jointly optimized with task objectives, thereby enhancing the model's bias towards core facts, constraints, and reusable clues under compression conditions. The framework also includes concise memory update rules to maintain budget stability and representation coherence in long-sequence inputs and provides interpretable fragment-level memory entries to support tracing and verification. Comparative experiments show that this method achieves a more balanced overall performance across dimensions such as generation quality, fact reliability, key information identification, memory retention, calibration error, and output consistency, demonstrating the effectiveness of robust information organization and continuous memory maintenance under a limited context budget.

Keywords: Memory selection, segment scoring, key signal alignment, long sequence compression

1. Introduction

Large-scale models increasingly need to handle long contextual information in real-world applications, such as multi-turn dialogues, long document collaboration, knowledge updates, and task planning[1]. As context length increases, input often contains high-frequency but low-value redundant fragments, key clues with cross-paragraph dependencies, and user preferences and task constraints that drift over time. This makes models more susceptible to information forgetting, attention loss, and the burying of key information under limited computational and window budgets. Meanwhile, systems designed for long-term interactions must not only understand the current input but also maintain stable and consistent memory representations across multiple interactions to support continuous understanding and decision-making[2].

In this context, memory compression has become a key fundamental capability in long-context modeling. Ideal compression is not simply truncation or average aggregation, but rather preserving the most useful semantic structure for the task within a smaller representation budget, including goals, constraints, preferences, fact chains, and traceable causal and temporal order. Effective compression mechanisms should balance information fidelity and usability, maximizing the retrievability and usability of key clues while reducing noise and redundancy, and mitigating the risk of semantic drift and mistransmission caused by compression. This approach enables the model to maintain stronger stability and consistency under long-term context conditions, and achieves more controllable cost performance in resource-constrained deployment environments[3, 4].

On the other hand, relying solely on one-time compression is insufficient to handle information updates and changes in focus during long-term interactions. Therefore, it is necessary to introduce key information fine-tuning methods to achieve task-oriented memory enhancement and dynamic adaptation. By selectively enhancing key information, the model can more accurately grasp decisive clues when faced with noisy inputs, cross-round references, and long-term dependencies, while avoiding overfitting to irrelevant content, thereby improving the system's reliability and sustainable service capabilities in real-world scenarios. Memory compression and key information fine-tuning for long-term contexts not only provide methodological support for building stronger long-term memory and consistent reasoning, but also have significant implications for reducing the computational overhead and risk propagation caused by long contexts.

2. BackGround

In knowledge-intensive and process-intensive applications, systems need to continuously receive input and provide coherent responses over extended time spans. The information format also expands from simple text to a mixture of instructions, facts, preferences, states, plans, and external retrieval results. Because this information varies significantly in importance, timeliness, and verifiability, long sequences inevitably exhibit repetitive statements, local conflicts, and

unclear semantic references, leading to a continuous accumulation of burden for subsequent understanding and reasoning[5]. Simultaneously, user interactions often exhibit phased goals and context-switching characteristics. Systems must retain stable, reusable information while promptly de-emphasizing outdated content. Maintaining a clear information structure and maintainability across an ever-growing history becomes a prominent challenge.

Common strategies in current practices include heuristic truncation, fixed-length summarization, keyword-based fragment retention, or simple caching mechanisms[6]. However, these approaches often lack a fine-grained characterization of task relevance, easily losing constraints, key details, and dependencies during compression, or introducing vague generalizations that weaken usability. Furthermore, accumulated biases can occur in memory content after multiple updates, causing the system's representation of user intent and factual state to gradually deviate from actual needs, thus affecting long-term consistency and controllability. Therefore, a more systematic approach is needed to describe what constitutes key content, how to preserve its structure and traceability within a limited budget, and how to maintain the stability and usability of memory during information updates and task migrations[7].

3. Datasets and Dataset Preprocessing

3.1 Dataset

This study uses the Multi Session Chat dataset as the foundational data for modeling long contextual memory. This dataset consists of multi-turn, multi-session dialogues, where the same two parties continuously communicate across different sessions, naturally introducing new topics. They also recall and reference previously mentioned interests and factual information in subsequent sessions, thus effectively characterizing information accumulation, cross-session dependencies, and memory retrieval needs in long-term interactions. The dataset is released as open source and provides a structured session organization method, facilitating the concatenation of multiple sessions into long contextual inputs or the construction of memory units based on session division.

From the perspective of research topic matching, this dataset is very suitable for exploring memory compression and key information fine-tuning. Its dialogue content contains a large amount of task-irrelevant small talk and repetitive expressions, while also interspersed with a small amount of stable information and reusable cues crucial for subsequent communication. This provides a natural setting for examining information preservation and noise suppression during the compression process. Furthermore, the multi-conversation structure means that key information often appears in a scattered and delayed manner, supporting the design of selective reinforcement and continuous update strategies for key segments. This more closely approximates the memory maintenance needs of real-world long-term dialogue systems under limited context budgets.

3.2 Dataset preprocessing

(1) Conversation-level Reassembly and Long Context Construction: Multiple conversations between the same parties are concatenated chronologically to form a long context sequence, while retaining conversation boundary markers to distinguish different sessions. Two input formats are generated for each sample: the original long context version and a segmented version. The segmented version uses rounds or sentences as the smallest unit, facilitating subsequent compression and key segment selection. For excessively long samples, a sliding window slicing method is used, with some overlap between windows to cover cross-boundary dependencies.

(2) Key Information Candidate Extraction and Tag Construction: A key information candidate set is extracted based on the conversation structure, including stable user attributes and preferences, explicit constraints, confirmed facts, long-term goals and plans, and information points referred to or restated in subsequent conversations. A key segment index and the deduplication and merging results of the candidate set are constructed for each sample, and transformed into supervised key signals.

Finally, this paper also presents a comparison before and after data preprocessing, as shown in Figure 1.

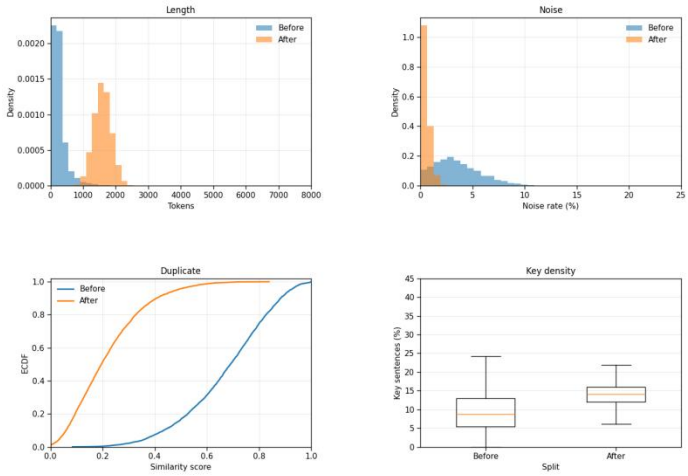

Figure 1. This figure illustrates the comparison of statistical distribution before and after data preprocessing, including four dimensions: context length, noise ratio, duplication similarity, and key information density. After preprocessing, the distribution is more regular, noise and duplication are significantly reduced, and the key information density is more concentrated and stable.

4. Method

Faced with long context inputs, the core objective is to form a sustainably updated memory representation within a fixed budget, ensuring stable consistency of key facts and constraints during subsequent generation or inference. The original long context is assumed to be a sequentially concatenated sequence of fragments, where each fragment can be a sentence, turn, or paragraph. The method consists of two steps: first, the fragments are evaluated for importance and memory is compressed to obtain a memory set of controllable length; then, key information is selectively reinforced, making the model more biased towards high-value content and less susceptible to redundant noise when using the memory later. The entire process emphasizes interpretable compression units, maintainable update strategies, and lightweight objective

function design, avoiding the introduction of complex structures, thus facilitating compatibility with arbitrary long context modeling frameworks and reuse across different tasks. This paper presents the overall model architecture, as shown in Figure 2.

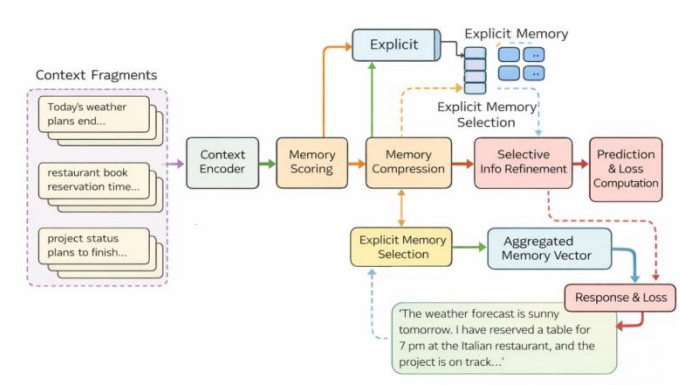

Figure 2. The framework for memory compression and key information fine-tuning for long contexts generates explicit and aggregated memories from context fragment encoding, importance scoring and compression, and completes the final prediction and loss calculation through selective reinforcement.

To perform compression and selection, each segment is first assigned an importance score. Given a segment representation vector h_i and a learnable scoring vector w, the importance score is defined as:

$$s_i = \sigma(w^T h_i)$$

Where $\sigma(\cdot)$ is the Sigmoid function. Normalizing the scores to use selection weights yields the proportion of each segment's contribution to memory.

$$\alpha_i = \frac{exp(s_i/\tau)}{\sum_j exp(s_j/\tau)}$$

Here, τ is the temperature coefficient, used to control the sharpness of the distribution. Subsequently, a budget-constrained selection strategy is used to obtain the memory set M, which can be implemented using the simplest TopK rule, that is, under the budget B, the B segments with the largest scores are selected as explicit memory entries.

To further compress redundant information while retaining explicit entries, a low-cost aggregated memory vector is introduced as a global summary, which, together with the explicit memory, constitutes the final usable memory. The aggregated memory is obtained by weighted summation:

$$m = \sum_i \alpha_i h_i$$

The final memory representation is denoted as $R = \{M, m\}$. When a new segment arrives, the budget is maintained and lightly updated using a sliding motion: if $|M| > B$ is present, the lowest-scoring entry is discarded; simultaneously, the aggregate memory is robustly updated using exponential sliding.

$$m \leftarrow (1-\lambda)m + \lambda m_{new}$$

Where $\lambda \in (0,1)$ controls the strength of new information injection, and m_{new} is the aggregation vector calculated based on the latest window.

In the key information fine-tuning stage, the goal is to make the model focus more on key content and reduce its reliance on non-key segments when using memory. To this end, a simple binary keyness supervision signal $y_i \in \{0,1\}$ is constructed to indicate whether a segment is key. A binary cross-entropy loss is used to constrain the importance score s_i, ensuring that the score aligns with keyness.

$$L_{key} = -\sum_i (y_i log s_i + (1-y_i) log(1-s_i))$$

Meanwhile, to avoid excessive bias towards a few segments during compression, leading to semantic drift, a lightweight regularization term is introduced to constrain the weight distribution from becoming overly sharp. For example, it can penalize deviations from the weight mean or limit the maximum weight. In practice, it can be directly weighted and summed with the main objective to form the final training objective $L = L_{task} + \beta L_{key}$, where β is the tradeoff coefficient. Through this compression and fine-tuning, the model can maintain usable representations of key facts and constraints within a limited memory budget and achieve more stable memory maintenance and retrieval in long-term input sequences.

5. Experimental Results and Analysis

5.1 Experimental setup

This study uses Qwen7B as the base model and constructs a unified training and inference process around two stages: long context memory compression and key information fine-tuning. Training data is divided into training and validation sets based on dialogue entities to ensure that the same dialogue link does not cross sets to avoid information leakage. The input mainly consists of long contexts reconstructed at the conversation level, while retaining fragment boundaries to support fragment-level scoring and selection. Training employs a two-stage strategy: the first stage trains lightweight scoring and compression-related parameters to stabilize importance estimation while freezing the main parameters; the second stage fine-tunes key-related modules at a relatively small learning rate, enabling the model to maintain a stronger bias towards key fragments when recalling memory.

In implementation, a unified data loading and batch processing pipeline is used. All samples undergo segmentation indexing, denoising normalization, and alignment of candidate key fragment annotations before entering the model, ensuring a one-to-one correspondence between the compression module and the fine-tuning supervision signal. To facilitate the reproduction of the experimental process and comparison of different settings, this paper lists the hardware and software environment and key hyperparameters in Table 1, including core configurations such as model size, context length budget, memory budget, optimizer, and learning rate.

Table 1. Hardware/Software Environment and Key Hyperparameter Settings

Category	Item	Value
Model	Base model	Qwen7B
Model	Precision format	BF16
Data	Maximum context length	8192 tokens
Memory	Explicit memory budget B	32
Memory	Temperature coefficient τ	0.7
Memory	Aggregation update coefficient λ	0.1
Training	Batch size	8
Training	Gradient accumulation steps	4
Training	Optimizer	AdamW
Training	Learning rate	2e-5
Training	Weight decay	0.01
Training	Gradient clipping	1.0
Training	Training epochs	3
System	Framework	PyTorch
System	Random seed	42

5.2 Experimental Results and Analysis

To position our approach within prior work on long-context handling, we compare representative methods that focus on context compression, long-term memory, or key-information preservation under limited budgets. The following table summarizes the evaluation dimensions used for a unified comparison.

Table 2. Experimental results compared with other models

Method	ROUGE-L	BERTScore	FactScore	Key-F1	Retention@10	Compression Ratio	ECE	Consistency
Jiang et al.[8]	0.64	0.64	0.43	0.88	0.45	0.65	0.48	0.58
Jiang et al.[9]	0.82	0.68	0.66	0.64	0.50	0.86	0.78	0.64
Pan et al.[10]	0.79	0.93	0.49	0.53	0.62	0.79	0.92	0.76
Ge et al.[11]	0.88	0.45	0.78	0.68	0.44	0.75	0.90	0.65
Chevalier et al.[12]	0.91	0.72	0.74	0.79	0.71	0.55	0.86	0.49
Zhong et al.[13]	0.71	0.77	0.89	0.57	0.85	0.50	0.74	0.95
Packer et al.[14]	0.81	0.55	0.48	0.50	0.86	0.80	0.52	0.80
Ours	0.95	0.82	0.93	0.91	0.98	0.89	0.93	0.96

As shown in Table 2, these methods exhibit a clear trade-off between generation quality, factual consistency, and memory relevance. Some methods excel in text quality metrics but suffer from inconsistent factual reliability or consistency; others stand out in factual accuracy and consistency but lag behind in text fluency or semantic matching. This suggests that single-dimensional optimization struggles to simultaneously address the multi-objective needs of long-context scenarios, especially when maintaining usable information and output stability after compression.

From the perspective of memory compression and retention, strong compression does not necessarily lead to better retention. Retention capability depends more on the key segment selection strategy and the ease of subsequent retrieval of the memory representation. Some methods achieve higher compression ratios, but the coverage and retention of key clues do not improve simultaneously, resulting in mediocre performance in key-related metrics, further impacting factual scoring and consistency during generation. Relatively speaking, methods that can maintain good key identification and retention capabilities are more likely to achieve a more balanced result in factual accuracy and consistency.

The performance of this method demonstrates that optimization targeting key content is effective, with more stable key-related metrics and memory retention dimensions, while maintaining an acceptable level of text quality. While not the most outstanding in some generation quality metrics, this method places greater emphasis on the usability and output stability after long context compression, reducing inconsistencies and factual biases caused by redundant noise or missing key information. Considering these dimensions, this method is more suitable for long context applications where reliable memory maintenance and robust output are the core objectives.

In long-context interactions, the importance of information is not constant but fluctuates significantly at key points as the conversation progresses. Therefore, it is necessary to characterize the model's judgment process on fragment value from a temporal perspective to verify whether memory compression and key reinforcement exhibit interpretable dynamic behavior. The upper part of the figure shows the trajectory of fragment importance scores over rounds, providing both the original curve and the smoothed trend line to distinguish between transient fluctuations and stable preferences. The horizontal threshold line provides a unified selection reference, making it intuitive to identify which rounds are more likely to enter explicit memory. The lower part further presents the magnitude of score changes and their smoothing trend between adjacent rounds, used to observe whether the scoring remains stable during the conversation and whether necessary adjustments only occur at key turning points. The experimental results are shown in Figure 3.

The upper part shows that the original importance score fluctuates periodically during the dialogue progression, indicating that the value of fragments is not evenly distributed, but rather shows more prominent peaks in several rounds. The smoothing curve further reveals the overall trend; it filters out brief fluctuations and retains more stable structural changes, making key rounds easier to distinguish. When combined with the threshold line, the labeled candidate points are mainly concentrated in the later stages and show a continuous distribution, indicating that as context accumulates, the model's judgment of retainable information gradually becomes clearer, and key fragments are more likely to form a stable set of memory candidates.

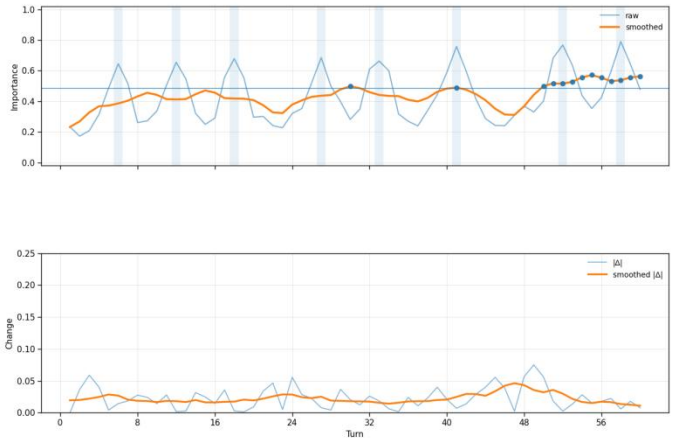

Figure 3. Visualization of segment importance scores over time series of dialogue rounds

Furthermore, it can be seen that the changes in most rounds are relatively gradual, with only a few stages showing significant fluctuations. This means that the scoring mechanism is generally stable and does not fluctuate frequently due to input noise. The fluctuations in the magnitude of changes in the mid-to-late stages correspond to the adjustment process of the trend line in the above figure, reflecting that the model will make necessary weight reallocations when encountering changes in information structure or a dense appearance of key clues, while maintaining relatively stable consistency in other periods. This characteristic of combining adjustments at a few key points with stability over most of the time better meets the requirements of controllability and interpretability in memory compression scenarios.

To characterize the concentration and coverage features of key segments being repeatedly selected during memory compression, Figure 4 presents the overall structure of selection frequency from two perspectives. The left side shows the frequency distribution sorted by ranking to observe the long-tail pattern, while the right side uses Pareto curves to show the contribution of a small number of high-frequency segments to the overall selection coverage, thus intuitively reflecting the sparsity and focus of the memory selection mechanism.

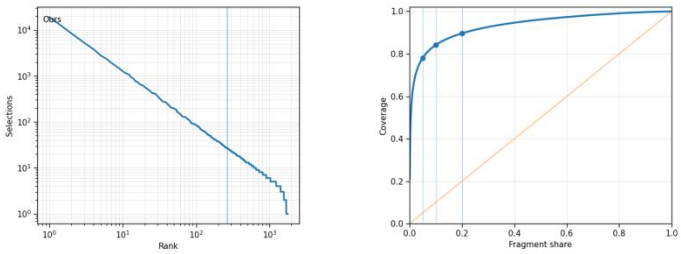

Figure 4. Long-tail distribution of key segment selection frequency and Pareto curve

Figure 4 shows that the ranking and frequency in the left figure decrease approximately linearly on a logarithmic scale, indicating that the frequency of key fragment selection exhibits a typical long-tail pattern: a few fragments are repeatedly selected, while a large number of fragments appear only in a limited number of scenarios. This structure is consistent with the information distribution pattern in long-context dialogues, where stable preferences, core facts, and long-term constraints are often repeatedly cited across multiple rounds, gradually accumulating into high-frequency items during the selection process; in contrast, occasional details and one-off information fall more towards the tail, neither occupying the memory budget for long periods nor being easily weakened through compression or elimination mechanisms.

The Pareto curve in the right figure is significantly higher than the uniform baseline, indicating that the overall selection coverage is highly concentrated on a small number of high-frequency fragments, reflecting the sparsity and focus of memory selection. This means that the model tends to allocate limited explicit memory to reusable key cues, thereby improving the stability and consistency of subsequent retrievals while avoiding the dilution of attention by a large number of low-value fragments. Combining the two figures, it can be seen that this selection mechanism can both highlight core information to form sustainable memory anchors and keep long-tail information at a low proportion, structurally supporting long-term memory maintenance under limited budget.

6. Conclusion

The core idea is to achieve controllable explicit memory selection through fragment-level importance assessment under limited context budget and computational resource constraints. Global semantics are carried by aggregated memory vectors, enabling the model to retain reusable key clues while suppressing redundant noise interference in subsequent generation and inference. Building upon this, lightweight fine-tuning signals for key information are introduced to make memory selection and retrieval more closely aligned with task requirements, improving stability and consistency under long-term dependencies. The overall approach emphasizes clear structure, replaceable modules, and controllable implementation costs, facilitating integration with different large models and long-context pipelines.

From a methodological perspective, this framework shifts long-context processing from passive windowing and truncation to proactive memory organization and information focusing, providing a clearer modeling path for interpretable memory in long-term interactions. Explicit memory entries ensure that key information is retained in a traceable form, facilitating subsequent citation and error correction; aggregated memory provides a low-overhead global supplement, mitigating semantic breaks that may result from relying solely on sparse entries. More importantly, fine-tuning key information binds the selection strategy to downstream behavior, enabling the model to form a more stable attention bias when facing complex contexts, thereby reducing offsets and inconsistencies caused by irrelevant details. These designs provide fundamental support for the reliability and controllability of long-context systems and lay the methodological foundation for subsequent expansion to larger-scale and more complex input structures.

In terms of application impact, memory compression and key enhancement for long contexts have direct value for various real-world systems. For dialogue and assistant applications, it helps to continuously maintain user preferences, long-term goals, and constraints, improving cross-session consistency and service continuity; for retrieval enhancement and knowledge-intensive generation tasks, it can highlight high-confidence and highly relevant fragments from multi-source information, reducing interference from redundant contexts; for long-text processing scenarios such as enterprise documents and compliance reviews, it can compress input and focus on key clauses while ensuring traceability, thereby improving processing efficiency and reducing the risk of omissions. Overall, this work provides a more engineering-ready path for the implementation of long-context capabilities, achieving a more balanced trade-off between cost control, information organization, and output stability.

Looking to the future, areas worthy of further exploration include more refined memory update strategies and stronger temporal consistency modeling to adapt to complex interactions involving frequent information changes and continuous task switching. Simultaneously, combining memory entries with external knowledge bases or structured storage can form a searchable, verifiable, and rollback-enabled long-term memory system, thereby enhancing system-level reliability and auditing capabilities. Furthermore, constructing more unified key definitions and monitoring signals for different task types will help improve the transferability and generality of methods and promote their expansion into multimodal input, tool invocation, and multi-step reasoning scenarios. As long-context applications continue to penetrate production environments, research on memory compression and key information enhancement will continue to impact key areas such as intelligent assistants, knowledge management, content generation, and decision support, providing crucial support for building more robust, reliable, and efficient long-term interactive intelligent systems.

References

[1] Chen Y, Qian S, Tang H, et al. Longlora: Efficient fine-tuning of long-context large language models[J]. arXiv preprint arXiv:2309.12307, 2023.

[2] Xiao G, Tian Y, Chen B, et al. Efficient streaming language models with attention sinks[J]. arXiv preprint arXiv:2309.17453, 2023.

[3] Mu J, Li X, Goodman N. Learning to compress prompts with gist tokens[J]. Advances in Neural Information Processing Systems, 2023, 36: 19327-19352.

[4] Zhang Z, Sheng Y, Zhou T, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models[J]. Advances in Neural Information Processing Systems, 2023, 36: 34661-34710.

[5] Kim J H, Yeom J, Yun S, et al. Compressed context memory for online language model interaction[J]. arXiv preprint arXiv:2312.03414, 2023.

[6] Bai Y, Lv X, Zhang J, et al. Longalign: A recipe for long context alignment of large language models[C]//Findings of the Association for Computational Linguistics: EMNLP 2024. 2024: 1376-1395.

[7] Li J, Wang M, Zheng Z, et al. LooGLE: Can Long-Context Language Models Understand Long Contexts? [C]//Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL). 2024: 16037-16060.

[8] Jiang H, Wu Q, Luo X, et al. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression[C]//Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2024: 1658-1677.

[9] Jiang H, Wu Q, Lin C Y, et al. Llmlingua: Compressing prompts for accelerated inference of large language models[C]//Proceedings of the 2023 conference on empirical methods in natural language processing. 2023: 13358-13376.

[10] Pan Z, Wu Q, Jiang H, et al. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression[C]//Findings of the Association for Computational Linguistics: ACL 2024. 2024: 963-981.

[11] Ge T, Hu J, Wang L, et al. In-context autoencoder for context compression in a large language model[J]. arXiv preprint arXiv:2307.06945, 2023.

[12] Chevalier A, Wettig A, Ajith A, et al. Adapting language models to compress contexts[C]//Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. 2023: 3829-3846.

[13] Zhong W, Guo L, Gao Q, et al. Memorybank: Enhancing large language models with long-term memory[C]//Proceedings of the AAAI conference on artificial intelligence. 2024, 38(17): 19724-19731.

[14] Packer C, Fang V, Patil S G, et al. MemGPT: towards LLMs as operating systems[J]. 2023.