R^{3}Mem: Bridging Memory Retention and Retrieval via Reversible Compression

Xiaoqiang Wang[1,2], **Suyuchen Wang**[1,2], **Yun Zhu**[3*] **and Bang Liu**[1,2,4†]
[1]DIRO & Institut Courtois, Université de Montréal
[2]Mila - Quebec AI Institute; [3]Google; [4]Canada CIFAR AI Chair
{xiaoqiang.wang, suyuchen.wang, bang.liu}@umontreal.ca, yunzhu@google.com

Abstract

Memory plays a key role in enhancing LLMs' performance when deployed to real-world applications. Existing solutions face trade-offs: explicit memory designs based on external storage require complex management and incur storage overhead, while implicit memory designs that store information via parameters struggle with reliable retrieval. In this paper, we propose **R^{3}Mem**, a memory network that optimizes both information **R**etention and **R**etrieval through **R**eversible context compression. Specifically, R^{3}Mem employs virtual memory tokens to compress and encode infinitely long histories, further enhanced by a hierarchical compression strategy that refines information from document- to entity-level for improved assimilation across granularities. For retrieval, R^{3}Mem employs a reversible architecture, reconstructing raw data by invoking the model backward with compressed information. Implemented via parameter-efficient fine-tuning, it can integrate seamlessly with any Transformer-based model. Experiments demonstrate that our memory design achieves state-of-the-art performance in long-context language modeling and retrieval-augmented generation tasks. It also significantly outperforms conventional memory modules in long-horizon interaction tasks like conversational agents, showcasing its potential for next-generation retrieval systems.

1 Introduction

Large language models (LLMs) (Ouyang et al., 2022; Team et al., 2023; Dubey et al., 2024) have demonstrated remarkable capabilities in natural language understanding and generation (Liang et al., 2022; Srivastava et al., 2023; Wang et al., 2024a), achieving human-comparable performance on complex reasoning tasks (Guo et al., 2023; Suzgun and Kalai, 2024). Deploying LLMs as controllers

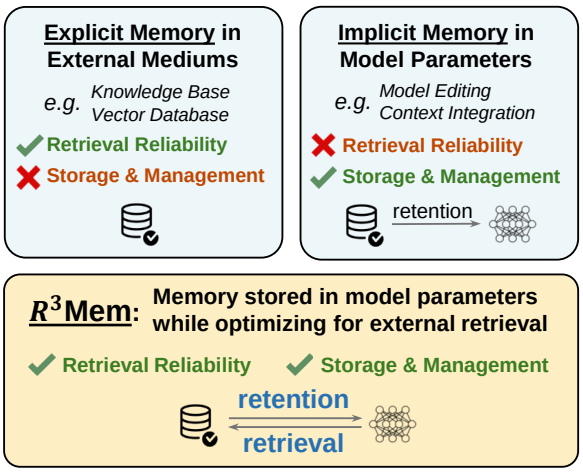

Figure 1: Comparison between explicit memory, implicit memory, and our proposed R^{3}Mem memory design.

to interact with dynamic environments and solve real-world tasks, *i.e.*, as autonomous agents (Liu et al., 2025), has shown promising success across diverse applications, including conversational assistants (OpenAI, 2022; Achiam et al., 2023), workflow automation (Hong et al., 2023; Wu et al., 2024; Wang and Liu, 2024; Qin et al., 2025), and embodied navigation (Wang et al., 2023a; Zheng et al., 2024; Sun et al., 2024b).

However, LLMs have inherent limitations: their stateless nature (Sumers et al., 2023) makes them struggle with leveraging past experiences for multi-turn interactions and cross-task generalization. Furthermore, their reliance on fixed context windows and static parameterized knowledge constrains their ability to handle complex tasks requiring dynamic, up-to-date information (Tao et al., 2024).

To address these challenges, existing approaches introduce external storage (*i.e.*, explicit memory), such as knowledge repositories (Kagaya et al., 2024; Zhu et al., 2024b) and vector databases (Liu et al., 2024; Jing et al., 2024), to enhance long-term retention and enable cross-task generalization (Maharana et al., 2024; Wang et al., 2023a,b) and cross-model sharing (Gao and Zhang, 2024). In parallel,

*Work done while at Google.
†Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2025, pages 4541–4557
July 27 - August 1, 2025 ©2025 Association for Computational Linguistics

implicit memory encodes contextual information directly into model parameters, enabling continuous knowledge updates while providing a more compact representation of information, reducing redundancy compared to external storage. Model-editing methods modify neurons to update (Huang et al., 2023; Gangadhar and Stratos, 2024) or forget knowledge (Wang et al., 2024d), while context integration (Choi et al., 2022; Wang et al., 2024c) adjusts internal parameters via model distillation. Memory-augmented Transformers (*e.g.*, RMT (Bulatov et al., 2022), Associate Memory (He et al., 2024; Wang et al., 2024b; Tack et al., 2024), and Titans (Behrouz et al., 2024)) enhance retention by integrating dedicated memory components.

However, as illustrated in Figure 1, both explicit and implicit memory involve trade-offs between storage overhead and recall effectiveness. Explicit memory grows indefinitely, requiring complex memory management techniques such as merging (Yin et al., 2024; Hu et al., 2024) and forgetting (Zhong et al., 2024). In contrast, implicit memory suffers from unreliable retrieval due to the black-box nature of LLMs, leading to confabulation and hallucination issues (Li et al., 2024a). As analyzed by Padmanabhan et al. (2024), injected atomic facts can propagate and influence broader inferences, further complicating retrieval accuracy. More recently, adaptive retrieval (Mallen et al., 2023; Farahani and Johansson, 2024) and MemoRAG (Qian et al., 2024) combine explicit and implicit memory in a hybrid retrieval paradigm but remain dependent on large-scale external storage.

In this paper, we propose **R^{3}Mem**, a novel memory-augmented model that optimizes both **mem**ory **r**etention and **r**etrieval while minimizing external storage dependency. R^{3}Mem leverages a **r**eversible architecture that integrates context compression and expansion, enabling assimilation and reconstruction of input data.

Specifically, we design a context compression task that learns to generate compressed representations ('query') from raw input ('context'). R^{3}Mem utilizes virtual memory tokens to encode and retain text that is indefinitely long. To improve compression quality, we introduce a hierarchical compression strategy, progressively refining information at the document, paragraph, and entity levels.

For retrieval, R^{3}Mem adopts a reversible architecture, reconstructing raw input by inverting the model invocation on compressed representations. This is achieved through adapter tuning, allowing seamless integration with pre-trained Transformer model while maintaining parameter efficiency.

To optimize both memory retention and retrieval, we employ bidirectional training with cycle consistency. The forward process encodes context into compressed memory representations, while the backward process reconstructs the raw content from memory tokens, enforcing consistency between the original and reconstructed information.

We evaluate R^{3}Mem on memory-intensive tasks, achieving state-of-the-art performance in long-context language modeling and retrieval-augmented generation. We also integrate R^{3}Mem into a real-world conversational agent that requires long-horizon interactions and the ability to recall distant historical context. R^{3}Mem consistently outperforms existing memory modules, demonstrating superior scalability, retrieval accuracy, and potential for next-generation retrieval systems.

2 Methodology

In this section, we introduce R^{3}Mem, a memory network that optimizes both memory retention and retrieval. As illustrated in Figure 2, the core component of R^{3}Mem is *context compression*, which encodes raw text into model parameters using *virtual memory tokens*. These trainable tokens are appended to the raw text, summarizing the current context window and propagating information to subsequent windows. This enables the model to absorb and retain indefinitely long input sequences. Furthermore, to facilitate more flexible memory usage, *i.e.*, enabling retrieval of documents for queries with varying semantic granularities, we employ hierarchical compression. This approach chunks documents into multiple levels of semantic representation, including document-, paragraph-, sentence-, and entity-level abstractions. By structuring information hierarchically, our method optimizes retention and retrieval efficiency across different levels of granularity. Lastly, we use a pre-trained Transformer backbone with an adapter-based reversible architecture, allowing the memory network to operate bidirectionally. This allows the model to be invoked in reverse, which reconstructs raw information from compressed memory akin to a "zip" and "unzip" process, unifying information retention and retrieval within a duplex network.

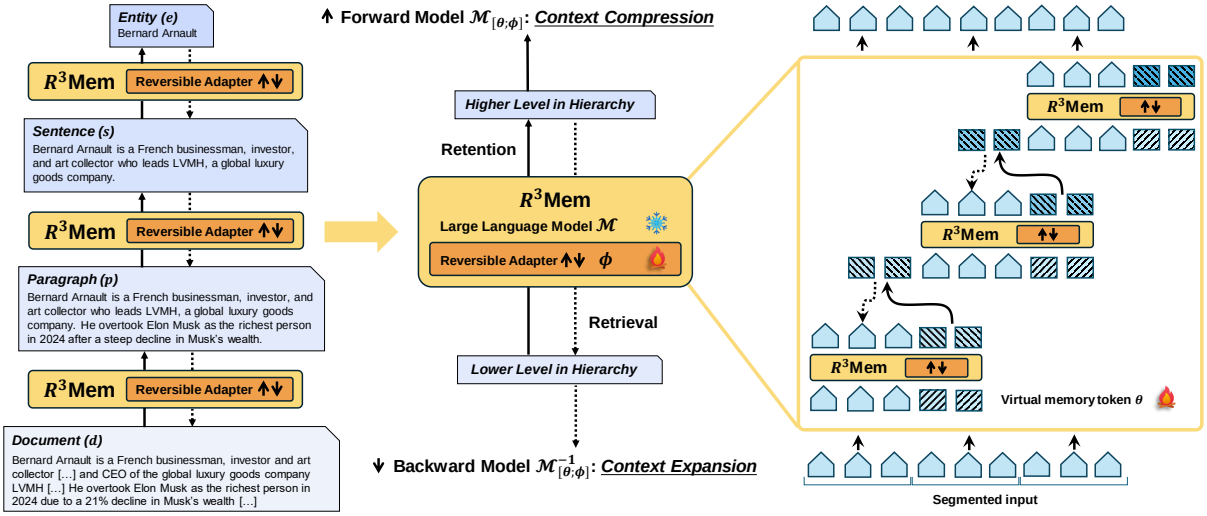

Figure 2: Overview of R^{3}Mem's architecture: The model employs a reversible framework that integrates context compression and expansion mechanisms. For the forward model, raw textual data is hierarchically encoded into compact representations at various levels—document, paragraph, and entity—using virtual memory tokens. In the backward model, the model reconstructs the original information by reversing the compression process.

2.1 Memory Retention

Inspired by context-supervised pretraining (Gao and Callan, 2022; W et al., 2023), which trains models to generate one passage conditioned on another from the same document, we employ a similar mechanism to bridge the information gap between condensed memory and raw content. Specifically, we formulate memory retention as a context compression problem, where the model learns to generate a compressed representation ('query' q) given a raw text input ('context' c).

To facilitate more flexible memory usage, we employ **hierarchical compression** to enhance multi-granularity assimilation, constructing $\langle c, q\rangle$ pairs at multiple levels, including *document-to-paragraph*, *paragraph-to-sentence*, and *sentence-to-entity* mappings. This structured approach segments documents into different semantic granularities, ensuring optimized retention and adaptive retrieval across varying levels of abstraction.

Furthermore, we introduce **virtual memory tokens** to efficiently encode long contexts by splitting them into manageable segments and processing them sequentially while preserving previous information. These tokens cache and propagate memory across context windows, ensuring continuity in long-context retention and enabling the model to maintain coherence over extended sequences.

Formally, given a context-query pair $\langle c, q\rangle$ from the context-query set D_c, the memory network $\mathcal{M}_\theta$ learns to model the conditional probability $\mathcal{M}_\theta(q \mid c)$ using an autoregressive decoder:

$$\mathcal{M}_\theta(q \mid c) = \prod_{t=1}^{T} \mathcal{M}_\theta(q_t \mid q_{<t}, c) \tag{1}$$

where T is the length of generated query comprised of a sequence of tokens $q = \langle q_1, \cdots, q_t, \cdots, q_T\rangle$, and θ denotes the virtual memory token.

Hierarchical compression. To construct a structured hierarchy of text chunks, we borrow pipeline from Xu et al. (2023) and Yoon et al. (2024a) to employ a superior LLM to decompose each document d into paragraphs p, sentences s, and key entities e (detailed in Section 3). At each level, the preceding granularity (*e.g.*, entire document) serves as the context and the subsequent (*e.g.*, paragraphs) as the query, forming structured $\langle c, q\rangle$ pairs:

$$D_c = D_d \cup D_p \cup D_s \tag{2}$$

$$= \{\langle d, p\rangle\}_1^N \cup \{\langle p, s\rangle\}_1^M \cup \{\langle s, e\rangle\}_1^K \tag{3}$$

Virtual memory tokens. Encoding lengthy contexts, such as long-term interaction histories with LLMs, often exceeds the model's effective context window. As analyzed by An et al. (2024), even with a theoretically large context length, a model's ability to retain and effectively utilize relevant information remains limited in practice. A naive approach would be to split long documents into smaller segments and process them individually. However, this disrupts semantic continuity and results in suboptimal training, as segmentation fragments contextual dependencies.

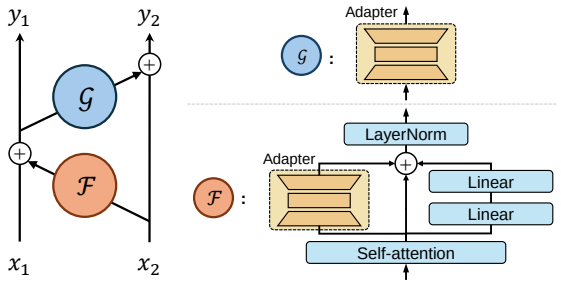

Figure 3: The architecture of the reversible Transformer. Left: The general reversible neural architecture. Right: The components of the reversible Transformer.

To address this, we introduce virtual memory tokens to bridge representations across context windows. These tokens act as summary vectors, caching compressed representations and transferring them across context windows. Formally, as shown in Figure 2, given a long input sequence c and segmented as $c = c^1 \oplus c^2 \cdots \oplus c^s \cdots \oplus c^S$. we prepend and append memory tokens to each segment as: $c^s = \theta^r \oplus c^s \oplus \theta^w$, where $\oplus$ denotes concatenation, θ^r represents the memory token outputs from the previous segment (with the c^1 having no such input), serving as memory *read* tokens for the current segment, and θ^w represents the memory tokens of the current segment, acting as memory *write* tokens to summarize the current segment and store information for future segments. By leveraging virtual memory tokens, the model can scale beyond context length limitations while maintaining continuity across segments.

Although token-based compression techniques have been widely explored in global attention (Zaheer et al., 2020; Beltagy et al., 2020), our virtual memory tokens differ in that these tokens are trainable, enabling adaptive context compression and efficient optimization within a prompt-tuning paradigm (Lester et al., 2021; Liu et al., 2021). Moreover, they can further enhance memory capability by inserting memory tokens as hidden states within each Transformer layer (Li and Liang, 2021), as detailed in Section 3.4.

2.2 Memory Retrieval

Considering the dual nature of memory retention and retrieval, where retention integrates raw text into a compressed representation (*i.e.*, compressing context into memory) and retrieval reverses this process by reconstructing the compressed representation into raw content (*i.e.*, expanding memory to context), we propose building R^{3}Mem with a reversible architecture. By simply flipping the input and output ends, this approach enables a duplex transformation between context and its memory, allowing simultaneous optimization of memory retention and retrieval to improve retrieval accuracy.

As shown in Figure 3, reversible architectures are a class of neural networks based on NICE (Dinh et al., 2014, 2022), which construct nonlinear bijective transformations by partitioning input at each layer into two groups that cache information for one another, thereby allowing exact reconstruction of inputs. Since the standard Transformer architecture is not inherently reversible, Liao et al. (2024) introduced adapter-based modifications to pre-trained Transformers to make them reversible. The key idea is to treat the original Transformer layer as one input group and the inserted adapter module as another, forming a reversible Transformer where the adapters are optimized using adapter tuning (Houlsby et al., 2019; Hu et al., 2021). We provide a more detailed introduction of reversible Transformer in Appendix C.

We use the pre-trained Transformer-based LLaMA 3.1-8B as the base model and integrate adapter modules to enable a reversible architecture. This allows us to reconstruct the input by feeding the compressed content backward. Formally, we denote the flipped model as $\mathcal{M}^{-1}$, where the backward generation models a similar conditional probability as the forward process in Eq. 1:

$$\mathcal{M}^{-1}_{[\theta;\phi]}(c \mid q) = \prod_{l=1}^{L} \mathcal{M}^{-1}_{[\theta;\phi]}(c_l \mid c_{<l}, q) \quad (4)$$

where L is the length of the generated context, encompassing a sequence of tokens $c = \langle c_1, \cdots, c_l, \cdots, c_L \rangle$, and ϕ is the adapter matrix.

2.3 Training Objective

Following the standard training setup of reversible architectures (He et al., 2016; Zheng et al., 2021; Wu, 2023), we optimize R^{3}Mem through bidirectional training with cycle consistency, incorporating forward compression loss, backward expansion loss, and a cycle consistency loss.

$$\mathcal{L} = \mathcal{L}_{\text{forward}} + \mathcal{L}_{\text{backward}} + \lambda \mathcal{L}_{\text{cycle}} \quad (5)$$

where λ is the coefficient to balance the contribution of cycle consistency loss.

Given a context-query pair $\langle c, q \rangle \in D_c$, forward training optimizes the memory network to model the probabilities of forward generation, as defined

in Eq. 1, by minimizing the conditional negative log-likelihood (NLL) loss:

$$\mathcal{L}_{\text{forward}} = -\sum_{t=1}^{T} \log \widehat{\mathcal{M}}_{[\theta;\phi]}\left(q_t \mid q_{<t}, c\right) \quad (6)$$

where $\widehat{\mathcal{M}}_{[\theta;\phi]}\left(q_t \mid q_{<t}, c\right)$ represents the predicted probability for token q_t in the reference query.

Similarly, backward training models the probabilities of backward generation as defined in Eq. 4:

$$\mathcal{L}_{\text{backward}} = -\sum_{l=1}^{L} \log \widehat{\mathcal{M}}^{-1}_{[\theta;\phi]}\left(c_l \mid c_{<l}, q\right) \quad (7)$$

where $\widehat{\mathcal{M}}^{-1}_{[\theta;\phi]}\left(c_l \mid c_{<l}, q\right)$ denotes the predicted probability for c_l in the reconstructed context.

To ensure cycle consistency, given an input c, we generate its reconstruction $\bar{c}$ by passing it through the reversible model as the forward mapping $f_{\mathcal{M}}$ and backward mapping $f_{\mathcal{M}^{-1}}$:

$$f_{\mathcal{M}} : c \longmapsto q \quad (8)$$

$$f_{\mathcal{M}^{-1}} : q \longmapsto \bar{c} \quad (9)$$

The cycle consistency loss maximizes the similarity between the original input c and its reconstruction $\bar{c}$ using cross-entropy:

$$\mathcal{L}_{\text{cycle}} = -\sum_{l=1}^{L} \log \widehat{\mathcal{M}}^{-1}_{[\theta;\phi]}\left(c_l \mid c_{<l}, f_{\mathcal{M}}(c)\right) \quad (10)$$

3 Experiments

Dataset. We follow the training protocol of Qian et al. (2024) using UltraDomain (Qian et al., 2024), which includes documents from the training set of diverse long-context question-answering and summarization tasks, including NarrativeQA (Kočiský et al., 2018), Qasper (Dasigi et al., 2021), GovReport (Huang et al., 2021), and MultiNews (Fabbri et al., 2019). Following Xu et al. (2023); Yoon et al. (2024a), we employ a more capable LLM as an oracle to generate hierarchical context-query pairs. While certain lightweight doc2query models (Nogueira et al., 2019; W et al., 2023) demonstrate strong performance in constructing context-query pairs, they often struggle with longer inputs. In Section 3.4, we present an experimental comparison between context-query pairs generated by doc2query models and the oracle model.

Specifically, we prompt a high-capacity oracle model (*i.e.*, GPT-4o) to progressively decompose each document into paragraphs, sentences, and entities. This process draws inspiration from event-centric hierarchical summarization methods (Zhong et al., 2022; Zhu et al., 2024a). Firstly, given a document d, the oracle generates a set of query-worthy events and selects the most relevant entities. We then prompt the oracle to retrieve sentence-level contexts, *i.e.*, s, surrounding these entities and condense them into sentence-entity pairs $\langle s, e \rangle$. Building on these pairs, we instruct the oracle to extend and summarize the retrieved context into paragraph-level chunks, *i.e.*, p, to create document-paragraph pairs $\langle d, p \rangle$ and paragraph-sentence pairs $\langle p, s \rangle$. Finally, we apply a length-based criterion to filter out short paragraphs and sentences. Paragraphs shorter than 20% of the corresponding original document length and sentences shorter than 4% are removed. The statistics of these constructed pairs are presented in Table 4 and the used prompts are provided in Appendix B.

Baselines. We compare R^{3}Mem with five memory-augmented Transformer architectures, categorized into recurrent architectures and associative memory architectures. The former include RMT (Bulatov et al., 2022), MemoRAG (Qian et al., 2024), and MELODI (Chen et al., 2024), while the latter comprise MemoryLLM (Wang et al., 2024b) and CAMELoT (He et al., 2024). For RMT, MemoryLLM, and MemoRAG, we utilize their official implementations to report results. Since MELODI and CAMELoT have not publicly released their code, we report their results as presented in their respective papers and ensure that our evaluation settings align with theirs for a fair comparison. Implementation details are provided in Appendix A.

3.1 Retention Performance

We firstly demonstrate whether R^{3}Mem can effectively compress and encode context. Following the setting of MELODI (Chen et al., 2024), we assess retention performance by measuring perplexity in long-context language modeling across three publicly available datasets: PG19 (Rae et al., 2019) , Pile arXiv (Gao et al., 2020), and C4 (4K+) (Raffel et al., 2020). The detailed experimental setup is provided in Appendix B. The average perplexity on the testing set is summarized in Table 1.

R^{3}Mem achieves state-of-the-art performance in long-context modeling. R^{3}Mem attains the lowest perplexity across all three datasets, effectively compressing long contexts into memory vectors. Notably, on the challenging C4 (4K+) dataset, R^{3}Mem reduces the perplexity by approximately 13% com-

Model	PG19	arXiv	C4 (4K+)
MemoryLLM	7.65	4.00	18.14
CAMELoT	7.10	3.60	-
RMT	7.04	3.56	17.67
MELODI	6.21	-	15.25
MemoRAG	5.92	3.35	15.37
R^{3}Mem	**5.21**	**2.39**	**13.38**

Table 1: Long-context language modeling performance in terms of perplexity among three benchmarks. The dash "-" indicates that the code is not publicly available and corresponding results are not reported in their paper.

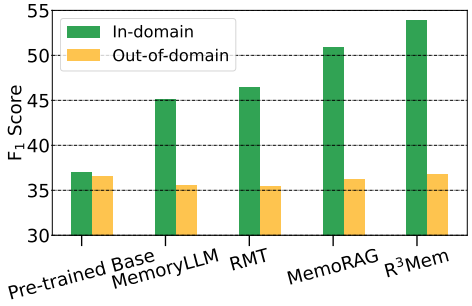

Figure 4: RAG performance on the UltraDomain dataset in terms of in-domain and out-of-domain settings.

pared to the next-best baseline, MemoRAG.

Recurrent architectures outperform associative memory. For example, associative memory-based methods, such as MemoryLLM and CAMELoT, exhibit inferior performance compared to the others. This disparity may stem from the reliance of associative memory approaches on discrete read and write operations, as seen in CAMELoT, and drop operations, as in MemoryLLM, which may struggle to maintain smoothly evolving contextual representations over long sequences.

3.2 Retrieval Performance

We further validate that the encoded information can be faithfully retrieved, establishing a reliable foundation for retrieval tasks. To assess this, we follow the experimental setup of MemoRAG (Qian et al., 2024) and integrate R^{3}Mem into a retrieval-augmented generation (RAG) question-answering (QA) task on UltraDomain, using the same in-domain and out-of-domain evaluation settings.

During evaluation, the model receives only the query as input, without direct access to the original test set context. This setup allows us to assess whether the model can effectively recall and utilize encoded context to generate accurate responses. The average F_1 Scores are shown in Figure 4. The results indicate that **retrieval performance is closely aligned with compression performance**, *i.e.*, better compression leads to improved retrieval. This demonstrates the dual nature of context compression and expansion, highlighting the rationale behind R^{3}Mem optimizing both to enhance retrieval accuracy. Notably, R^{3}Mem achieves the best results in both retrieval and retention performance, while MemoryLLM underperforms in both aspects.

Out-of-domain performance is significantly lower than in-domain performance, but remains consistent across models. The out-of-domain performance is notably lower than in-domain performance, as observed in R^{3}Mem's results (in-domain: 53 vs. out-of-domain: 36). In contrast, the base pre-trained model exhibits a smaller gap (in-domain: 38 vs. out-of-domain: 37), suggesting that fine-tuning on domain-specific data has minimal impact on out-of-domain generalization. This indicates that integrating new context does not degrade the model's original knowledge. Additionally, the performance differences across models in the out-of-domain evaluation are relatively small. R^{3}Mem achieves only a marginal 1% improvement over the weakest-performing MemoryLLM, and all baselines exhibit out-of-domain performance similar to the base model. To further investigate this phenomenon, we scale training iterations and analyze its effects in Section 3.4.

3.3 Agent Performance

We assess R^{3}Mem in a real-world agent data using SiliconFriend (Zhong et al., 2024), an AI chatbot companion. Specifically, we replace its external memory module, *i.e.*, MemoryBank (Zhong et al., 2024) with R^{3}Mem. We use the publicly available SiliconFriend dataset, which consists of interactions among 15 distinct virtual users over a 10-day period. Following the setup of Zhong et al. (2024), where the external memory bank is initialized with given dialogue history, we initialize the implicit memory module by training R^{3}Mem on this dialogue history for two epochs. For comparison, we employ MemoRAG (*i.e.*, the most competitive baseline in Table 1) as the baseline implicit memory module and fine-tune it under the same settings.

We use 194 memory-probing questions. First, the models retrieve context from the memory bank or generate context using MemoRAG and R^{3}Mem. The retrieved or generated context is then fed into SiliconFriend to generate final responses. The evaluation includes four key metrics: (1) *Memory Retrieval Accuracy*, measuring alignment with

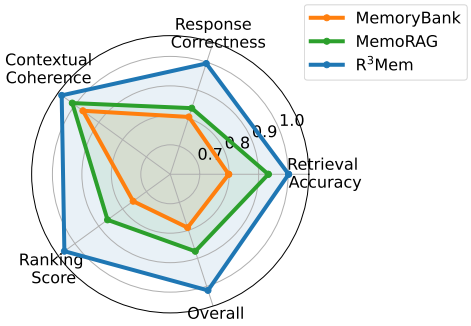

Figure 5: Evaluation of memory retrieval and response generation when integrating R^{3}Mem into the SiliconFriend conversational agent. The overall score represents the average across all four evaluation metrics. Scores are re-scaled using min-max normalization for each metric to enhance clarity.

Model	PG19	arXiv	C4 (4K+)
Fine-tuned Base	8.10	4.31	19.04
R^{3}Mem-context-only	7.15	3.69	17.16
R^{3}Mem-short-context	7.44	3.83	17.60
R^{3}Mem-w/o-backward	6.41	3.21	15.83
R^{3}Mem-w/o-cycle	5.91	2.87	14.80
R^{3}Mem	**5.21**	**2.39**	**13.38**

Table 2: Ablation analysis of R^{3}Mem on long-context language modeling task.

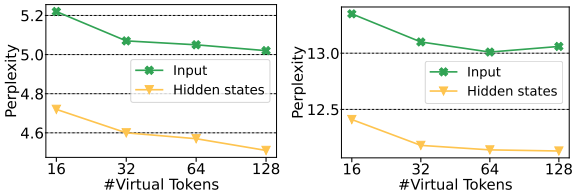

Figure 6: Long-context language modeling performance on the PG19 (left) and C4 (4K+) (right) dataset when scaling virtual memory tokens with increasing number and injection of hidden states.

the reference memory using the F_1 Score; (2) *Response Correctness*, assessing whether the response contains the correct answer via exact substring matching; (3) *Contextual Coherence*, evaluating response fluency and relevance using BARTScore-Faithfulness (Yuan et al., 2021); and (4) *Ranking Score*, ranking memory modules based on response correctness, with scores computed as $s = 1/r$, where $r \in \{1, 2, 3\}$ denotes ranking position.

The results, summarized in Figure 5, reveal two key findings. Firstly, **implicit memory modules outperform explicit memory**. For instance, both R^{3}Mem and MemoRAG surpass the original MemoryBank across all four metrics. Beyond *Ranking Scores* and *Response Correctness*, *Memory Retrieval Accuracy* exhibits the most significant difference between implicit memory modules and explicit memory. Notably, R^{3}Mem achieves the best overall performance, primarily due to its superior *Memory Retrieval Accuracy*. However, **Contextual Coherence shows no significant differences across memory modules**. This could be due to the fact that SiliconFriend has been fine-tuned on psychological dialogues, enabling it to generate fluent and natural responses even when the retrieved memory is not entirely accurate.

3.4 In-depth Analysis

Hierarchical compression and high-quality context-query pairs improve performance. We construct two baseline models that exclude hierarchical context-query pairs: (1) **R^{3}Mem-context-only**, which retains only document-paragraph pairs (*i.e.*, $\langle d, p \rangle$ in Eq. 3), and (2) **R^{3}Mem-short-context**, which generates short context-query pairs using a lightweight doc2query model (W et al., 2023). The latter feeds document chunks of fewer than 512 tokens as input and generates queries.

We train both baselines using same setting as R^{3}Mem. As shown in Table 2, removing hierarchical compression or restricting context-query length significantly increases perplexity. The short-context variant leads to an even larger performance drop, reducing retention effectiveness by approximately 28% on C4 (4K+). These results highlight the necessity of hierarchical compression for effectively encoding context of R^{3}Mem.

Injecting virtual tokens into hidden states improves performance, but increasing token length does not. We assess the impact of scaling virtual memory tokens: (1) increasing the number of input virtual tokens from 16 to 32, 64, and 128, and (2) injecting virtual tokens into the hidden states of each Transformer layer, following Li and Liang (2021), with scaling from 16 to 32, 64, and 128.

The results, illustrated in Figure 6, reveal two key observations. First, adding virtual tokens to hidden states improves final performance, suggesting that similar to hierarchical compression, hierarchical memory across Transformer layers enhances memory retention. However, this approach drastically increases the number of trainable parameters by a factor of 32 in the LLaMA 3.1-8B base model, making it impractical for large-scale deployment. As a result, R^{3}Mem defaults to using virtual tokens only in the input sequence. Second, increasing token length (*i.e.*, memory size) does not significantly improve performance, which is

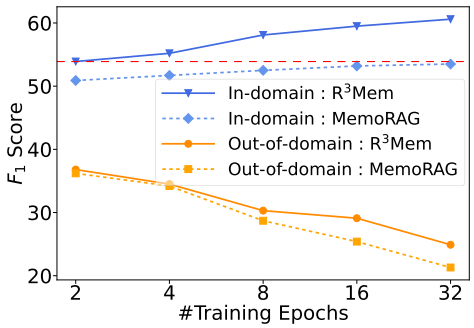

Figure 7: RAG performance when strengthening memory through training extra epochs on the training set.

inconsistent with prompt-tuning. A possible explanation is that memory tokens serve a different function from prompt tuning. While prompt tuning enhances task-level generalization by stimulating pre-trained knowledge, memory tokens summarize and store sample-level context, making them easier to fit than multi-task learning.

Backward optimization and cycle consistency loss are essential. We investigate the effect of training objectives by testing two ablated variants: (1) **R^{3}Mem-w/o-backward**, which removes the backward loss, and (2) **R^{3}Mem-w/o-cycle**, which omits cycle consistency loss. As shown in Table 2, both variants exhibit degraded performance, demonstrating that all three loss components contribute to optimal training by ensuring alignment between context compression and expansion. This alignment is key to R^{3}Mem, enabling memory retention and retrieval in a duplex framework.

Extra training improves in-domain retrieval at the cost of out-of-domain generalization. We analyze whether overfitting to the training set strengthens memorization of training contexts. We extend training from 2 epochs to 4, 8, 16 and 32 epochs and evaluate in-domain and out-of-domain retrieval performance. Results in Figure 7 indicate that while in-domain retrieval consistently improves with prolonged training, it comes at the cost of out-of-domain generalization, observed in both R^{3}Mem and MemoRAG. This suggests that although implicit memory avoids the management and storage overhead of explicit memory, integrating new memory through fine-tuning may be unstable, as newly encoded memory may interfere with or even overwrite pre-trained parametric knowledge, making it harder for effective lifelong integration.

R^{3}Mem demonstrates efficient memory retrieval, particularly under large memory volumes. To better assess practical usability, we further analyze the inference-time efficiency of each memory module by decomposing latency into memory construction (write) and retrieval (read) stages under increasing conversation history sizes (1K, 2K, and 5K turns). The evaluation follows the setting introduced in Section 3.3, and all results are measured using wall-clock time on the same hardware setup to ensure a fair comparison.

Module	Write (ms)	Read (ms)	Performance
1K history			
MemoryBank	**795**	**183**	72.8
MemoRAG	1476	986	75.9
R^{3}Mem (short)	876	367	83.9
R^{3}Mem	1539	366	**86.1**
2K history			
MemoryBank	**794**	514	72.3
MemoRAG	1469	1137	75.5
R^{3}Mem (short)	869	**376**	83.5
R^{3}Mem	1538	378	**86.4**
5K history			
MemoryBank	**792**	1453	69.5
MemoRAG	1466	1358	75.6
R^{3}Mem (short)	864	379	83.6
R^{3}Mem	1538	**360**	**85.9**

Table 3: Comparison of memory latency and performance across different history sizes. Performance is the overall score defined in Section 3.3. R^{3}Mem (short) denotes the variant that generates short context-query pairs using a lightweight doc2query model.

Explicit memory modules such as MemoryBank rely on dense retrieval over Faiss-indexed embeddings, resulting in low write latency (around 790 ms per record), but their read latency increases significantly with longer conversation histories, *i.e.* from 183 ms at 1K history to 1453 ms at 5K. In contrast, implicit memory modules like R^{3}Mem maintain stable read latency around 360–380 ms across all history sizes, as their decoding-based retrieval mechanism is independent of memory volume.

Although R^{3}Mem incurs higher write latency (approximately 1538 ms) due to LLM-based context compression, we also evaluate a variant, R^{3}Mem (short), where memory is constructed using a lightweight doc2query model. This variant reduces write latency to around 870 ms per record, while retaining most of the performance benefits.

With the full 5K-turn conversation history, R^{3}Mem achieves an average write and read time of 1536 ms and 360 ms respectively (totaling 1896 ms), which is notably lower than the 2245 ms required by MemoryBank (792 ms write + 1453 ms read). Moreover, R^{3}Mem attains a significantly higher overall performance score of 85.9 compared to MemoryBank's 69.5, demonstrating its clear ad-

vantage in both efficiency and effectiveness.

4 Related Works

Memory-augmented neural networks. Designing architectures capable of memorization and generalization through knowledge abstraction (Sukhbaatar et al., 2019) and data-dependent information retention (Zancato et al., 2024) has been a longstanding research focus. Early approaches introduced architectures with external memory modules, such as neural turing machines (NTM) (Graves, 2014) and modern Hopfield Networks (Ramsauer et al., 2020), which utilize pre-defined update rules to manage memory. With the advent of Transformers, some methods employ recurrent Transformer architectures (Dai, 2019; Bulatov et al., 2022) to cache key-value pairs as memory, enabling the reuse of cached information to extend context window sizes. Additionally, recent studies have explored encoding training data into model parameters, effectively using them as memory to store world knowledge (Wang et al., 2024c; Padmanabhan et al., 2024; Gangadhar and Stratos, 2024; He, 2024). This approach has also been extended to large databases (Qian et al., 2024), test-time data points (Sun et al., 2024a), and broader language modeling tasks (Yang et al., 2024). Titans (Behrouz et al., 2024) integrates long-term, short-term, and persistent memory into a unified neural architecture. While optimizing memory retention, they overlook retrieval reliability from model parameters, which is a core design motivation of R^{3}Mem.

Context compression. Compressing lengthy contexts into concise representations that retains essential information can make LLM inference more efficient (Choi et al., 2022; Li et al., 2024b). Approaches like Selective Context (Li et al., 2023), LLMLingua (Jiang et al., 2023, 2024) and RECOMP (Xu et al., 2023) use context selection to improve inference efficiency, and methods such as AutoCompressor (Chevalier et al., 2023), in-context autoencoder (ICAE) (Ge et al., 2023), Gist (Mu et al., 2024) and CompAct (Yoon et al., 2024b) employ training-based techniques to generate summary representations. Besides, Delétang et al. (2023) proposes new general-purpose language modeling perspectives by leveraging compression through arithmetic coding from information theory (Rissanen, 1976; Pasco, 1976). In contrast, R^{3}Mem uses context compression as a surrogate task to optimize memory retention while ensuring alignment through backward context expansion.

5 Conclusion

We propose R^{3}Mem, a memory network built on a reversible architecture that optimizes both information retention and retrieval. R^{3}Mem employs hierarchical compression to adaptively process input and utilizes virtual memory tokens to encode long-context information. Empirical results demonstrate state-of-the-art performance in long-context modeling and retrieval, with strong scalability and accuracy in real-world conversational agents.

Limitations

R^{3}Mem is a duplex network that unifies memory retention and retrieval, which learns to encode documents into model parameters through context compression. The limitations of this framework fall into two main aspects: (1) the trade-off between high-quality memory retention and the complexity and cost of the context-query construction pipeline, and (2) the instability of implicit memory in lifelong integration.

On the one hand, ensuring effective document assimilation requires hierarchical compression optimized through hierarchical context-query pairs. As analyzed in Section 3.4, the quality of these pairs significantly impacts memory retention effectiveness. For example, while a lightweight doc2query model produces reasonably good results, they are still less effective than those generated by more capable LLMs, which, in turn, come with significantly higher computational costs. Balancing high-quality memory retention with the complexity and cost of the context-query construction pipeline is crucial. Depending on the frequency of new context integration into the model parameters, incorporating an adaptive data construction pipeline within R^{3}Mem could enhance its efficiency, making this an important direction for future work.

On the other hand, strengthening memory retention through additional training may impact pretrained parametric knowledge or overwrite existing memory (as analyzed in Section 3.4). While this behavior may be desirable in narrow-domain applications or when maintaining a small-scale memory history, excessive training could undermine the model's inherent contextual understanding and commonsense reasoning. Further evaluation is needed to comprehensively understand these relationships. Additionally, developing a more con-

trollable memory architecture that better balances historical context retention with new knowledge integration, such as incorporating expert network routing mechanisms, remains an important avenue for future work on R^3Mem.

Ethics Statement

R^3Mem, as a memory network, can be integrated into memory-intensive applications (Zhang et al., 2024) such as social simulation, conversational assistants, and personalized recommendations. While the model can encode personal information from interaction history of applications into its parameters, its training-based nature allows for the filtering of harmful or sensitive content during memory construction. This ensures a personalized experience and optimal retrieval performance while safeguarding users from potential harm.

Acknowledgements

This work is supported by the Canada CIFAR AI Chair Program and the Canada NSERC Discovery Grant (RGPIN-2021-03115).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Chenxin An, Jun Zhang, Ming Zhong, Lei Li, Shansan Gong, Yao Luo, Jingjing Xu, and Lingpeng Kong. 2024. Why does the effective context length of llms fall short? *arXiv preprint arXiv:2410.18745*.

Ali Behrouz, Peilin Zhong, and Vahab Mirrokni. 2024. Titans: Learning to memorize at test time. *arXiv preprint arXiv:2501.00663*.

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

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

Yinpeng Chen, DeLesley Hutchins, Aren Jansen, Andrey Zhmoginov, David Racz, and Jesper Andersen. 2024. Melodi: Exploring memory compression for long contexts. *arXiv preprint arXiv:2410.03156*.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. 2023. Adapting language models to compress contexts. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3829–3846, Singapore. Association for Computational Linguistics.

Eunbi Choi, Yongrae Jo, Joel Jang, and Minjoon Seo. 2022. Prompt injection: Parameterization of fixed inputs. *arXiv preprint arXiv:2206.11349*.

Zihang Dai. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. 2021. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4599–4610, Online. Association for Computational Linguistics.

Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, et al. 2023. Language modeling is compression. *arXiv preprint arXiv:2309.10668*.

Laurent Dinh, David Krueger, and Yoshua Bengio. 2014. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2022. Density estimation using real nvp. In *International Conference on Learning Representations*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

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

Mehrdad Farahani and Richard Johansson. 2024. Deciphering the interplay of parametric and non-parametric memory in retrieval-augmented language models. *arXiv preprint arXiv:2410.05162*.

Govind Krishnan Gangadhar and Karl Stratos. 2024. Model editing by standard fine-tuning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 5907–5913, Bangkok, Thailand. Association for Computational Linguistics.

Hang Gao and Yongfeng Zhang. 2024. Memory sharing for large language model based agents. *arXiv preprint arXiv:2404.09982*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853, Dublin, Ireland. Association for Computational Linguistics.

Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2023. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*.

Alex Graves. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401*.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 820–828.

Xu Owen He. 2024. Mixture of a million experts. *arXiv preprint arXiv:2407.04153*.

Zexue He, Leonid Karlinsky, Donghyun Kim, Julian McAuley, Dmitry Krotov, and Rogerio Feris. 2024. Camelot: Towards large language models with training-free consolidated associative memory. *arXiv preprint arXiv:2402.13449*.

Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352*.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International conference on machine learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. 2024. Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model. *arXiv preprint arXiv:2408.09559*.

Luyang Huang, Shuyang Cao, Nikolaus Parulian, Heng Ji, and Lu Wang. 2021. Efficient attentions for long document summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1419–1436, Online. Association for Computational Linguistics.

Zeyu Huang, Yikang Shen, Xiaofeng Zhang, Jie Zhou, Wenge Rong, and Zhang Xiong. 2023. Transformer-patcher: One mistake worth one neuron. *arXiv preprint arXiv:2301.09785*.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13358–13376, Singapore. Association for Computational Linguistics.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677, Bangkok, Thailand. Association for Computational Linguistics.

Zhi Jing, Yongye Su, Yikun Han, Bo Yuan, Haiyun Xu, Chunjiang Liu, Kehai Chen, and Min Zhang. 2024. When large language models meet vector databases: A survey. *arXiv preprint arXiv:2402.01763*.

Tomoyuki Kagaya, Thong Jing Yuan, Yuxuan Lou, Jayashree Karlekar, Sugiri Pranata, Akira Kinose, Koki Oguri, Felix Wick, and Yang You. 2024. Rap: Retrieval-augmented planning with contextual memory for multimodal llm agents. *arXiv preprint arXiv:2402.03610*.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Johnny Li, Saksham Consul, Eda Zhou, James Wong, Naila Farooqui, Yuxin Ye, Nithyashree Manohar, Zhuxiaona Wei, Tian Wu, Ben Echols, et al. 2024a. Banishing llm hallucinations requires rethinking generalization. *arXiv preprint arXiv:2406.17642*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th*

International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4582–4597, Online. Association for Computational Linguistics.

Yucheng Li, Bo Dong, Frank Guerin, and Chenghua Lin. 2023. Compressing context to enhance inference efficiency of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 6342–6353, Singapore. Association for Computational Linguistics.

Zongqian Li, Yinhong Liu, Yixuan Su, and Nigel Collier. 2024b. Prompt compression for large language models: A survey. *arXiv preprint arXiv:2410.12388*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, et al. 2022. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*.

Baohao Liao, Shaomu Tan, and Christof Monz. 2024. Make pre-trained model reversible: From parameter to memory efficient fine-tuning. *Advances in Neural Information Processing Systems*, 36.

Bang Liu, Xinfeng Li, Jiayi Zhang, Jinlin Wang, Tanjin He, Sirui Hong, Hongzhang Liu, Shaokun Zhang, Kaitao Song, Kunlun Zhu, et al. 2025. Advances and challenges in foundation agents: From brain-inspired intelligence to evolutionary, collaborative, and safe systems. *arXiv preprint arXiv:2504.01990*.

Di Liu, Meng Chen, Baotong Lu, Huiqiang Jiang, Zhenhua Han, Qianxi Zhang, Qi Chen, Chengruidong Zhang, Bailu Ding, Kai Zhang, et al. 2024. Retrievalattention: Accelerating long-context llm inference via vector retrieval. *arXiv preprint arXiv:2409.10516*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. 2024. Evaluating very long-term conversational memory of llm agents. *arXiv preprint arXiv:2402.17753*.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Jesse Mu, Xiang Li, and Noah Goodman. 2024. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, et al. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Shankar Padmanabhan, Yasumasa Onoe, Michael Zhang, Greg Durrett, and Eunsol Choi. 2024. Propagating knowledge updates to lms through distillation. *Advances in Neural Information Processing Systems*, 36.

Richard Clark Pasco. 1976. *Source coding algorithms for fast data compression*. Ph.D. thesis, Stanford University CA.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

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

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. 2025. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. 2020. Hopfield networks is all you need. *arXiv preprint arXiv:2008.02217*.

Jorma J Rissanen. 1976. Generalized kraft inequality and arithmetic coding. *IBM Journal of research and development*, 20(3):198–203.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, and Armand Joulin. 2019. Augmenting self-attention with persistent memory. *arXiv preprint arXiv:1907.01470*.

Theodore R Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L Griffiths. 2023. Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, et al. 2024a. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv preprint arXiv:2407.04620*.

Zhiyuan Sun, Haochen Shi, Marc-Alexandre Côté, Glen Berseth, Xingdi Yuan, and Bang Liu. 2024b. Enhancing agent learning through world dynamics modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3534–3568, Miami, Florida, USA. Association for Computational Linguistics.

Mirac Suzgun and Adam Tauman Kalai. 2024. Meta-prompting: Enhancing language models with task-agnostic scaffolding. *arXiv preprint arXiv:2401.12954*.

Jihoon Tack, Jaehyung Kim, Eric Mitchell, Jinwoo Shin, Yee Whye Teh, and Jonathan Richard Schwarz. 2024. Online adaptation of language models with a memory of amortized contexts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zhengwei Tao, Ting-En Lin, Xiancai Chen, Hangyu Li, Yuchuan Wu, Yongbin Li, Zhi Jin, Fei Huang, Dacheng Tao, and Jingren Zhou. 2024. A survey on self-evolution of large language models. *arXiv preprint arXiv:2404.14387*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xing W, Guangyuan Ma, Wanhui Qian, Zijia Lin, and Songlin Hu. 2023. Query-as-context pre-training for dense passage retrieval. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1906–1916, Singapore. Association for Computational Linguistics.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023a. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Xiaoqiang Wang and Bang Liu. 2024. Oscar: Operating system control via state-aware reasoning and re-planning. *arXiv preprint arXiv:2410.18963*.

Xiaoqiang Wang, Lingfei Wu, Tengfei Ma, and Bang Liu. 2024a. FAC^{2}E: Better understanding large language model capabilities by dissociating language and cognition. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 13228–13243, Miami, Florida, USA. Association for Computational Linguistics.

Yu Wang, Yifan Gao, Xiusi Chen, Haoming Jiang, Shiyang Li, Jingfeng Yang, Qingyu Yin, Zheng Li, Xian Li, Bing Yin, Jingbo Shang, and Julian McAuley. 2024b. Memoryllm: towards self-updatable large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Yu Wang, Xinshuang Liu, Xiusi Chen, Sean O'Brien, Junda Wu, and Julian McAuley. 2024c. Self-updatable large language models with parameter integration. *arXiv preprint arXiv:2410.00487*.

Yu Wang, Ruihan Wu, Zexue He, Xiusi Chen, and Julian McAuley. 2024d. Large scale knowledge washing. *arXiv preprint arXiv:2405.16720*.

Zihao Wang, Shaofei Cai, Anji Liu, Yonggang Jin, Jinbing Hou, Bowei Zhang, Haowei Lin, Zhaofeng He, Zilong Zheng, Yaodong Yang, et al. 2023b. Jarvis-1: Open-world multi-task agents with memory-augmented multimodal language models. *arXiv preprint arXiv:2311.05997*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xianchao Wu. 2023. Duplex diffusion models improve speech-to-speech translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8035–8047, Toronto, Canada. Association for Computational Linguistics.

Zhiyong Wu, Chengcheng Han, Zichen Ding, Zhenmin Weng, Zhoumianze Liu, Shunyu Yao, Tao Yu, and Lingpeng Kong. 2024. Os-copilot: Towards generalist computer agents with self-improvement. *arXiv preprint arXiv:2402.07456*.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. 2023. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv preprint arXiv:2310.04408*.

Hongkang Yang, Zehao Lin, Wenjin Wang, Hao Wu, Zhiyu Li, Bo Tang, Wenqiang Wei, Jinbo Wang, Zeyun Tang, Shichao Song, et al. 2024. Memory^3: Language modeling with explicit memory. *arXiv preprint arXiv:2407.01178*.

Zhangyue Yin, Qiushi Sun, Qipeng Guo, Zhiyuan Zeng, Qinyuan Cheng, Xipeng Qiu, and Xuanjing Huang. 2024. Explicit memory learning with expectation maximization. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16618–16635, Miami, Florida, USA. Association for Computational Linguistics.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024a. Compact: Compressing retrieved documents actively for question answering. *arXiv preprint arXiv:2407.09014*.

Chanwoong Yoon, Taewhoo Lee, Hyeon Hwang, Minbyul Jeong, and Jaewoo Kang. 2024b. CompAct: Compressing retrieved documents actively for question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 21424–21439, Miami, Florida, USA. Association for Computational Linguistics.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.

Luca Zancato, Arjun Seshadri, Yonatan Dukler, Aditya Golatkar, Yantao Shen, Benjamin Bowman, Matthew Trager, Alessandro Achille, and Stefano Soatto. 2024. B'mojo: Hybrid state space realizations of foundation models with eidetic and fading memory. *arXiv preprint arXiv:2407.06324*.

Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501*.

Duo Zheng, Shijia Huang, Lin Zhao, Yiwu Zhong, and Liwei Wang. 2024. Towards learning a generalist model for embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13624–13634.

Zaixiang Zheng, Hao Zhou, Shujian Huang, Jiajun Chen, Jingjing Xu, and Lei Li. 2021. Duplex sequence-to-sequence learning for reversible machine translation. *Advances in Neural Information Processing Systems*, 34:21070–21084.

Ming Zhong, Yang Liu, Suyu Ge, Yuning Mao, Yizhu Jiao, Xingxing Zhang, Yichong Xu, Chenguang Zhu, Michael Zeng, and Jiawei Han. 2022. Unsupervised multi-granularity summarization. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4980–4995, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19724–19731.

Mengna Zhu, Kaisheng Zeng, Mao Wang, Kaiming Xiao, Lei Hou, Hongbin Huang, and Juanzi Li. 2024a. Eventsum: A large-scale event-centric summarization dataset for chinese multi-news documents. *arXiv preprint arXiv:2412.11814*.

Yuqi Zhu, Shuofei Qiao, Yixin Ou, Shumin Deng, Ningyu Zhang, Shiwei Lyu, Yue Shen, Lei Liang, Jinjie Gu, and Huajun Chen. 2024b. Knowagent: Knowledge-augmented planning for llm-based agents. *arXiv preprint arXiv:2403.03101*.

A Implementation Details

We developed our method using PyTorch (Paszke et al., 2019). We initialize the base model, LLaMA 3.1-8B, with checkpoints from the Hugging Face Transformers package (Wolf et al., 2020). We implemented the adapter module using LoRA (Hu et al., 2021), setting the scaling factor $\alpha = 32$ and the rank $r = 8$, with a dropout of 0.1 applied. The default setting for the number of virtual memory tokens is 8, unless scaled to 16, 32, 64, or 128, as discussed in Section 3.4. These tokens are randomly initialized by sampling from $\mathcal{N}(0, 0.02)$. We set the coefficient of the cycle consistency loss λ in Eq. 5 to 0.5. Fine-tuning is performed for 2 epochs using the AdamW optimizer (Loshchilov and Hutter, 2018) with a maximum learning rate of 2×10^{-5}, $\beta_1 = 0.9$, $\beta_2 = 0.99$, and a learning rate warmup period covering 6% of the total training steps. In Section 3.4, we scale the training epochs to 4, 8, 16, and 32. The batch size is 2, and training is conducted on a single NVIDIA RTX A5000 24 GB GPU, taking approximately 13 hours for a 2-epoch run. Experiments were conducted over four independent runs with different random seeds, and the best evaluation results were selected for reporting. For RMT, MemoryLLM, and MemoRAG, we utilize their official implementations to report results. For MELODI and CAMELoT, as their code is not publicly available, we report their results as stated in their respective papers and ensure that our evaluation settings align with theirs for a fair comparison.

B Experiment Setup

Context-query pairs. The prompt used to construct the hierarchical context-query pairs is presented in Figure 8. We use a low temperature of 0.3 and greedy decoding to preserve accurate event-related details. The statistics of the constructed context-query pairs are summarized in Table 4.

Long context language modeling. We firstly demonstrate whether R^{3}Mem can effectively compress and encode context. Specifically, following the setting of MELODI (Chen et al., 2024), we assess compression performance by measuring perplexity in long-context language modeling across three publicly available datasets: PG19 (Rae et al., 2019) , arXiv from the Pile (Gao et al., 2020), and C4 (4K+) (Raffel et al., 2020).

While MELODI also evaluates on a custom-collected dataset from arXiv Bulk Data Access, details about their data pipeline and cleaned data are not publicly available. Instead, we utilize the arXiv subset from the Pile dataset (Gao et al., 2020), which comprises technical papers in mathematics, computer science, and physics, totaling 1,264 documents in the test split.

	Document	Paragraph	Sentence	Entity
#Samples	2,178	10,198	50,989	152,968
Max. Length	9,528	1,803	42	12
Min. Length	1,356	207	364	1
Avg. Length	7,470	1,537	319	3

Table 4: Breakdown of statistics for the constructed context-query pairs. Length indicates the length of each text chunk, measured in tokens.

The PG19 test set includes 100 English books, each containing 68,972 tokens on average. For the C4 dataset, a web-crawled corpus of internet documents, we employ the "c4/en" subset, which has undergone cleaning and deduplication. To focus on long-context scenarios, we filter out samples with fewer than 4,096 tokens, resulting in 155,007 testing samples.

Retrieval-augmented generation. We further validate whether the encoded memory can be faithfully retrieved, establishing a reliable foundation for retrieval tasks. To assess this, we follow the experimental setup of MemoRAG (Qian et al., 2024) and integrate R^{3}Mem into a retrieval-augmented generation (RAG) question-answering (QA) task on UltraDomain, using the same in-domain and out-of-domain evaluation settings.

For in-domain evaluation, we use a subset of the UltraDomain test set, where both the training and test samples are based on the same underlying world knowledge. This knowledge is sourced from Wikipedia, research papers in S2ORC (Lo et al., 2020), ebooks from Project Gutenberg[1], and domain-specific financial and legal documents. For out-of-domain evaluation, we use another subset of the UltraDomain test set, where queries and contexts are drawn from textbooks spanning 18 diverse domains, including biology, religion, art, etc. This evaluation measures the model's ability to retrieve and apply knowledge that was not explicitly present in the training data, testing its generalization beyond the training distribution.

Conversational agent. We use 194 memory-probing questions. First, the models retrieve context from the memory bank or generate context using MemoRAG and R^{3}Mem. The retrieved or gen-

[1] https://www.gutenberg.org/

erated context is then fed into SiliconFriend to generate final responses.

The evaluation covers four key metrics: **(1) Memory retrieval accuracy**: It measures the alignment of retrieved memory with reference memory using the F_1-score. **(2) Response correctness**: It evaluates whether the response contains the correct answer. Since the gold answer may be embedded within a longer dialogue response, correctness is determined using exact substring matching. **(3) Contextual coherence**: It assesses whether the response is natural and coherent within the given context and dialogue history. This is evaluated using BARTScore-Faithfulness (Yuan et al., 2021), a widely used automatic metric for natural language generation that measures the relevance of the candidate response to the reference dialogue history. **(4) Model ranking score**: For each test question, the three memory modules—original memory bank, R^{3}Mem, and MemoRAG—are ranked based on response correctness. The models' scores are calculated as $s = 1/r, r \in \{1, 2, 3\}$, which indicates their ranking position.

C More Details About Reversible Transformers

In this section, we provide a more detailed explanation of reversible Transformers.

Reversible neural networks (Dinh et al., 2014, 2022) are constructed so that each layer's outputs suffice to exactly reconstruct its inputs. As shown in Figure 3, a common paradigm is to split the input of layer l into two groups, x_l^1 and x_l^2. Let the layer apply functions $\mathcal{F}_l$ and $\mathcal{G}_l$, producing outputs:

$$\mathbf{y}_l^1 = \mathbf{x}_l^1 + \mathcal{F}_l\left(\mathbf{x}_l^2\right) \tag{11}$$

$$\mathbf{y}_l^2 = \mathbf{x}_l^2 + \mathcal{G}_l\left(\mathbf{y}_l^1\right) \tag{12}$$

Because $\mathbf{y}_l^1$ and $\mathbf{y}_l^2$ can be inverted as

$$\mathbf{x}_l^1 = \mathbf{y}_l^1 - \mathcal{F}_l\left(\mathbf{x}_l^2\right) \tag{13}$$

$$\mathbf{x}_l^2 = \mathbf{y}_l^2 - \mathcal{G}_l\left(\mathbf{y}_l^1\right) \tag{14}$$

the forward transformation is bijective. Consequently, no intermediate activations beyond $\mathbf{y}_l^1$ and $\mathbf{y}_l^2$ need be stored in memory since intermediate states $\mathbf{x}_l^1, \mathbf{x}_l^2$ are fully recoverable in backward passes.

However, standard Transformers (Vaswani et al., 2017) use residual connections and sub-layer stacks that do not conform to these precise invertibility requirements. To address this, Liao et al. (2024) propose making Transformer layers reversible with lightweight modifications. Each Transformer block is divided into two functional "streams": (1) the original sub-layer (attention or feed-forward), augmented with an adapter module (Houlsby et al., 2019), and (2) a second stream that is an adapter-only module. The two streams act on two separate inputs $\mathbf{x}_l^1, \mathbf{x}_l^2$, arranged in a reversible pattern (analylogous to $\mathcal{F}_l$ and $\mathcal{G}_l$ above). One input passes through the original (frozen) Transformer sub-layer plus an adapter, while the other goes through a purely adapter-based function. The original Transformer parameters remain fixed, and only the adapters are trained. Because each stream can invert the other's output, the entire layer is fully reversible without re-training from scratch.

In our work, R^{3}Mem uses the reversible Transformer structure to simultaneously learn *context compression* in the forward pass and *context expansion* in the backward pass. By design, each layer allows exact reconstruction of its input. Thus, when we pass compressed representations "backward" through the network, we can recover the original text context. This bijective mechanism directly enforces consistency between memory retention (compression) and memory retrieval (expansion), enabling us to optimize both objectives together. Furthermore, exact invertibility minimizes activation storage and avoids the large overhead typically required to handle forward and backward passes in a standard Transformer.

Role: You are an advanced language model specializing in hierarchical summarization.
Task Overview: Given a document, your goal is to decompose the content step by step into events, entities, sentences, and paragraphs. This process involves:
1. Identifying key events and the most relevant entities.
2. Gathering sentence-level contexts around those entities.
3. Constructing paragraph-level summaries from those sentence-level contexts.
4. Produce the final output as nested JSON that follows the structure.

1. **Identify Events & Select Entities**
    - Scan the document to produce a set of query-worthy events.
    - For each event, choose the key entities (people, places, organizations, and concepts) most relevant to that event.
2. **Gather Sentence-Level Context & Form Sentence-Entity Pairs**
    - For each entity, locate the sentences in which the entity appears or is crucially described.
    - Create Sentence-Entity pairs: each pair references a sentence and the corresponding entity.
3. **Summarize into Paragraphs & Link Document-to-Paragraph / Paragraph-to-Sentences**
    - Group thematically related sentences together into paragraph-level summaries.
    - Produce document-paragraph pairs and paragraph-sentence pairs.
4. **Final Output as Pure-String Hierarchical JSON**
    - Maintain a strictly nested structure:
    - Entities must be a single string (e.g., “EntityA,EntityB”).
    - Ensure valid JSON syntax (quoted keys, values, and commas in the correct places).

Example of Final JSON Structure (All Values as Strings):

```
{
  "document_id": "d1",
  "paragraphs": [
    {
      "paragraph_id": "p1",
      "sentences": [
        {
          "sentence_id": "s1",
          "text": "Full text or condensed version of the first sentence.",
          "entities": "EntityA,EntityB"
        },
        {
          "sentence_id": "s2",
          "text": "Full text or condensed version of the second sentence.",
          "entities": "EntityA"
        }
      ]
    },
    {
      "paragraph_id": "p2",
      "sentences": [
        {
          "sentence_id": "s3",
          "text": "Full text or condensed version of the third sentence.",
          "entities": "EntityC,EntityD"
        }
      ]
    }
  ]
}
```

Figure 8: The prompt used to instruct GPT-4o to construct hierarchical context-query pairs in Section 3.